
**UNIVERSIDAD EUROPEA DE MADRID**

**ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO**

**GRADO EN INGENIERÍA INFORMÁTICA**


PROYECTO FIN DE GRADO

# ADMINISTRADOR DE CONTRASEÑAS BASADO EN BLOCKCHAIN


**YAGO IGLESIAS DÍAZ**

**Dirigido por**

**ALFONSO VILCHEZ DE LAS HERAS**


**CURSO 2023-2024**

**TÍTULO**: ADMINISTRADOR DE CONTRASEÑAS BASADO EN BLOCKCHAIN

**AUTOR**: YAGO IGLESIAS DÍAZ

**TITULACIÓN**: GRADO EN INGENIERÍA INFORMÁTICA

**DIRECTOR/ES DEL PROYECTO**: ALFONSO VILCHEZ DE LAS HERAS

**FECHA**: MAYO de 2024

# RESUMEN

Keychain es una aplicación web que interactúa con una blockchain privada para gestionar las contraseñas de los usuarios con las mejores prácticas de cifrado existentes, teniendo como principal objetivo el incremento de seguridad, privacidad y transparencia que se halla en esta industria.

El proyecto surge de un creciente aumento de la inseguridad digital y con ello una aceptación por parte de los usuarios a usar este tipo de soluciones, sin embargo, estas no aportan la transparencia suficiente, desencadenando desconfianza para gestionar las “llaves” que desbloquean toda la información personal de un individuo.

La aplicación desarrollada se adentra en el problema ofreciendo un sistema descentralizado e inmutable mediante el almacenamiento de las contraseñas en la blockchain, haciendo que sea posible acceder de manera segura a todas las contraseñas registradas recordando únicamente una que desbloquea las demás.

**Palabras clave:** Blockchain, descentralización, criptografía, contraseñas, clave privada, aplicación web.

# ABSTRACT

Keychain is a web application that interacts with a private blockchain to manage user passwords using the best existing encryption practices, with the main objective being to increase security, privacy, and transparency within this industry.

The project arises from a growing increase in digital insecurity and with it, a user acceptance of using these types of solutions. However, these do not provide sufficient transparency, triggering distrust in managing the "keys" that unlock all of an individual's personal information.

The developed application delves into the problem by offering a decentralized and immutable system through storing passwords on the blockchain, making it possible to securely access all registered passwords by only remembering one that unlocks the others.

**Keywords:** Blockchain, decentralization, cryptography, passwords, private key, web application.

***AGRADECIMIENTOS***

*Quiero expresar mi agradecimiento a mi tutor, Alfonso Vilchez de las Heras, por su orientación y apoyo durante la realización de este Trabajo Fin de Grado. Su experiencia y consejos han sido esenciales para el desarrollo de este proyecto.*

*Agradezco también a la familia, pareja y amigos por su feedback y opiniones realizadas en base al proyecto, para así permitir mejoras en la calidad de este. Además, a todos aquellos que han leído la memoria voluntariamente con el objetivo de ayudar a la mejora de redacción de esta.*

*En un entorno cada vez más digital y abstraídos de los sistemas que procesan nuestra información personal, es necesario poder brindar una solución que gestione contraseñas proporcionando la seguridad y transparencia que los usuarios requieren.*

# TABLA RESUMEN

| | **DATOS** |
|---|---|
| **Nombre y apellidos:** | YAGO IGLESIAS DÍAZ |
| **Título del proyecto:** | Administrador de Contraseñas basado en Blockchain |
| **Directores del proyecto:** | ALFONSO VILCHEZ DE LAS HERAS |
| **El proyecto ha implementado un producto:** | SI |
| **Objetivo general del proyecto:** | Desarrollar una aplicación web para gestionar contraseñas almacenadas en una blockchain. |

# Índice

# Índice de Figuras

# Índice de Tablas

# Capítulo 1. ANTECEDENTES / ESTADO DEL ARTE

## 1.1 Introducción

En el mundo digital actual, la seguridad y privacidad cada vez juegan un papel más importante en nuestras vidas. Los ataques cibernéticos están a la orden del día, siendo cada vez una preocupación más latente en la población.

La gran mayoría de las personas no toman las medidas necesarias para hacer frente a los distintos riesgos que existen en el mundo de la informática, siendo una de las principales causas la desinformación globalizada en este tema.

Por suerte, la concienciación en ciberseguridad está en auge, y con ello, han estado apareciendo estos últimos años distintas plataformas y sistemas, con el objetivo de ayudar a los usuarios de internet a tener más seguridad, como por ejemplo, las aplicaciones de gestores de contraseñas, donde un usuario puede almacenar sus distintas contraseñas accediendo así a un nivel superior de seguridad en la red.

Las aplicaciones que administran las contraseñas tradicionalmente están desarrolladas por multinacionales, lo cual puede llevar a inconvenientes como veremos más adelante. También, existen una variedad de tipos de estas plataformas, siendo las más seguras las que se ejecutan localmente en un servidor u ordenador “privado”, y las más cómodas y menos seguras las que se localizan en la nube, pudiendo acceder a ellas mediante un navegador convencional.

Este proyecto pretende mejorar y dar otro enfoque a estas plataformas, y para ello se han considerado los aspectos más relevantes para su desarrollo;

- La **seguridad** de las contraseñas es la prioridad principal del proyecto. Se deben considerar las diferentes técnicas de cifrado y mecanismo de protección para garantizar la confidencialidad e integridad de los datos.
- La **descentralización** utilizando una blockchain, permitiendo un almacenamiento distribuido, eliminando un punto central de fallo y mejorando la seguridad.
- La **usabilidad**, donde se busca mejorar aspectos como la interfaz y la experiencia de usuario para abstraer al cliente de la complejidad de la blockchain, y con ello garantizar que la aplicación sea fácil de usar y navegar.
- La **privacidad** de los usuarios y el control total de sus contraseñas debe ser respetada ante todo.

## 1.2 Estado del arte

El proyecto trata sobre el desarrollo de una aplicación web para gestionar las contraseñas de los usuarios, de manera descentralizada y segura mediante el uso de la tecnología blockchain y la criptografía.

### 1.2.1 Gestor de contraseñas

Un gestor de contraseñas es una plataforma que permite a los usuarios almacenar sus distintas contraseñas que posee con el fin de no tener que recordar ni preocuparse de ninguna de ellas, es decir, cuando el usuario necesite la contraseña, la consultará en la plataforma, consiguiendo así poder acceder a una contraseña más segura.

El contexto actual de estas aplicaciones en auge, ha hecho que actualmente exista una gran variedad de estas plataformas desarrolladas por empresas líderes, como pueden ser Google, Apple o Microsoft. (Fernández, Y., 2024)

Un estudio de Security.org donde se estudiaron los hábitos de más de 1.000 estadounidenses, señalaba este último año un crecimiento de la industria, reportando un aumento de un 60% de la población que utilizan gestores de contraseñas, es decir, una de cada tres personas utilizan gestores de contraseñas. Se estima que son 79 millones de usuarios. (Security.org, 2023)

Otros datos interesantes sobre la industria, recopilados por Norton, un antivirus de los más utilizados a nivel personal, son los siguientes: (Norton, 2023)

- El 28% de los que no usan gestores de contraseñas creen que son inseguros. (Security.org, 2023)
- Casi dos tercios de los usuarios de internet llevan un registro de sus contraseñas de memoria o con notas manuscritas. (Security.org, 2023)
- Más de tres cuartas partes de las personas cambiaron la forma en que gestionaban sus contraseñas después de ser afectadas por la toma de control de una cuenta. (Ponemon Institute, 2020)
- El 47% de los millennials son más propensos a memorizar sus contraseñas. (LastPass, 2022)
- Casi el 85% de los usuarios de internet que usan un gestor de contraseñas lo usan en su teléfono. (Security.org, 2023)
- Casi el 70% de los usuarios de gestores de contraseñas usan gestores de contraseñas gratuitos. (Security.org, 2023)

### 1.2.2 Criptografía

La criptografía es el ámbito que ocupa técnicas de cifrado o codificado destinadas a alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos incomprensibles a receptores no autorizados.

Este campo de estudio ha acompañado a la humanidad durante prácticamente toda su existencia, no obstante, a partir de mediados de los años 70, a causa de un desbloqueo de una versión de esta tecnología más avanzada por parte de la Agencia de Seguridad Nacional (NSA) de los Estados Unidos, empezó a generalizarse su uso para convertirse en como la conocemos hoy en día, en una práctica irremplazable, la cual se usa en todos los ámbitos del mundo de la informática, desde los navegadores usados comúnmente por los consumidores, hasta el almacenamiento de contraseñas de todas las plataformas. (Wikipedia, 2021)

Dentro de la criptografía, existen distintas disciplinas que son usadas para objetivos diferentes.

***Funciones hash***

Las funciones hash son algoritmos matemáticos que toman una entrada (como un archivo o un mensaje) y producen una salida única y de longitud fija, llamada hash. Estas funciones son unidireccionales, lo que significa que es fácil calcular el hash a partir de la entrada, pero es extremadamente difícil (casi imposible) reconstruir la entrada a partir del hash. El algoritmo actualmente de los más seguros y utilizados es **SHA-256**, habiendo resistido durante años a pruebas específicas que se realizan sobre las funciones hash para buscar sus vulnerabilidades.

***Cifrado simétrico***

El cifrado simétrico es un método de encriptación donde tanto el emisor como el receptor utilizan la misma clave para cifrar y descifrar los mensajes. Es rápido y eficiente. En este tipo de disciplina, el algoritmo de cifrado **AES** es el estándar por el gobierno de los Estados Unidos siendo de los más utilizados por su seguridad, ya que para intentar descifrar un mensaje de este algoritmo por fuerza bruta teniendo una clave de encriptación de 128 bits (16 caracteres), con el ordenador más potente del mundo, se tardaría 1 billón de billones de años. (EETimes, 2012)

En muchos casos, para hacer este tipo de cifrado más seguro se utiliza un componente llamado IV, el cual es una cadena de caracteres aleatoria que se concatena al mensaje que se quiere cifrar, esto se realiza para que aunque dos mensajes sean iguales, no tengan el mismo resultado al cifrarlos.

***Key Derivation Function (KDF)***

Una función de derivación de clave (KDF, por sus siglas en inglés) es un componente criptográfico que toma una clave inicial o una contraseña y la transforma en una clave más larga, más compleja o simplemente en otra clave que pueda ser más adecuada para su uso en un sistema criptográfico particular. Las KDF son usadas normalmente por los gestores de contraseñas, para, a partir de una clave maestra del usuario, obtener una cadena de caracteres aparentemente aleatoria y con más seguridad para cifrar mensajes.

En esta función, al igual que ocurre con el cifrado simétrico, se suele utilizar un componente llamado “salt”, que es una cadena de caracteres aleatoria que se concatena a la clave que se quiere derivar, esto se realiza para que aunque dos claves sean iguales, no tengan el mismo resultado al pasarlos por esta función.

Por lo tanto, el uso de determinados algoritmos de la criptografía hace intratable descifrar un mensaje para un actor externo sin la información necesaria, y con ello, se protege la privacidad de los mensajes.

### 1.2.3 Blockchain

La blockchain, a pesar de tener una connotación negativa para muchas personas por relacionarla con especulación y algunas criptomonedas de dudosa legitimidad, tiene detrás de ella una tecnología innovadora y con mucho potencial.

Es una tecnología de registro distribuido que permite la creación de bases de datos compartidas y seguras. Funciona como un libro de contabilidad digital que registra de manera transparente y permanente las transacciones entre diferentes partes sin necesidad de un intermediario central. Cada bloque de información está enlazado a los anteriores, formando una cadena inmutable, de ahí su nombre.

El primer blockchain fue conceptualizado por una persona (o grupo de personas) conocida como Satoshi Nakamoto en 2008, aunque lo hizo partiendo ya de distintas plataformas y tecnologías existentes como HashCash (sistema que usa Proof of Work para evitar los emails spam) o la criptografía, la propiedad más notable e importante de blockchain.

Más enfocados en el contexto actual, la industria de blockchain está aquí para quedarse, y tanto las empresas como las personas están comenzando a adoptarla. Zippia, una empresa estadounidense dedicada al mundo del empleo, expone algunas estadísticas clave sobre blockchain y criptomonedas:

- El gasto mundial en soluciones blockchain ascendió a $6.6 mil millones en 2021. (Zippia, 2023)
- Se proyecta que las empresas gastarán casi $19 mil millones sobre la tecnología blockchain en 2024. (Zippia, 2023)
- Sobre 300 millones de personas, 3.9% de la población mundial usa blockchain para criptomonedas. (Zippia, 2023)
- El 90% de EE.UU., los bancos canadienses y europeos han comenzado a explorar la tecnología blockchain. (Zippia, 2023)
- Hay más de 82 Millones de carteras de Bitcoin en el mundo. (Zippia, 2023)

La tecnología, junto con una adopción de esta por parte de empresas y usuarios, dan una perspectiva del sector prometedora de cara a un futuro cercano.

***Proof of Work***

Un término muy sonado en el mundo de la blockchain es el "proof of work" o prueba de trabajo, el cual es un concepto utilizado en blockchain para validar y asegurar transacciones. Implica que los participantes en la red deben realizar cálculos computacionales intensivos para demostrar que han dedicado una cantidad significativa de recursos computacionales a la validación de transacciones. Esto ayuda a prevenir ataques maliciosos y asegura la integridad de la red.

### 1.2.4 Ciberdelincuencia

La ciberdelincuencia es un término que abarca todas las actividades delictivas que se realizan utilizando medios electrónicos o digitales. Esto puede incluir una amplia gama de actividades, como el robo de información personal o financiera, el fraude en línea, la piratería informática, el acoso cibernético, la distribución de malware, entre otros.

Estos ataques están aumentando según evoluciona la tecnología, debido a que los ciberdelincuentes cada vez tienen más medios e incentivos para realizar estos ataques. Una investigación por parte del FBI de los reportes sobre los ataques en internet, expusieron un récord de $10.2 mil millones de pérdidas en ciberdelitos en el año 2022. (Federal Bureau of Investigation [FBI], s.f.) (Kolesnikov, N. 2023)

### 1.2.5 Situación de las contraseñas

La situación en la que se encuentra la seguridad de las contraseñas en la actualidad es muy preocupante, teniendo datos que demuestran la escasa protección que los usuarios dan a sus contraseñas.

Estas son algunas estadísticas recopiladas por Norton: (Norton, 2023)

- Los usuarios de internet que no utilizan gestores de contraseñas tienen tres veces más probabilidades de ser afectados por el robo de identidad. (Security.org, 2023)
- La contraseña más común es "123456". (Reader's Digest, 2023)
- El 18% de las personas usan el nombre de su mascota en sus contraseñas. (Security.org, 2023)
- El 13% de las personas usa la misma contraseña para todas sus cuentas. (Google, 2019)
- Más del 80% de las brechas confirmadas están relacionadas con contraseñas robadas, débiles o reutilizadas. (LastPass, 2021)
- El 96% de las contraseñas más comunes pueden ser descifradas por herramientas de hacking en menos de un segundo. (Digital Shadows, 2022)
- El 21% de las personas admite incluir su año de nacimiento en su contraseña. (Security.org, 2023)
- Casi tres cuartas partes de los que han intentado adivinar la contraseña de alguien han acertado. (Beyond Identity, 2021)

*(Estas son sólo unas estadísticas seleccionadas, se pueden consultar más en el anexo)*

Por otro lado, las estadísticas indican que tener una contraseña poco segura aumenta el riesgo cuando una empresa sufre una brecha de seguridad. Aunque las contraseñas filtradas estén hasheadas y no se vean en texto claro, si una contraseña es débil, es fácil probar unas pocas combinaciones hasta encontrar la correcta, obteniendo el mismo hash.

Esto se puede evitar utilizando contraseñas seguras. Un gestor de contraseñas es de gran ayuda en este sentido, ya que permite crear contraseñas muy complejas sin necesidad de recordarlas, solo consultarlas. Esto eleva exponencialmente el tiempo que un actor malicioso necesita para "crackear" la clave. En la siguiente figura, se puede apreciar el tiempo que se tarda en averiguar una contraseña.

| Number of Characters | Numbers Only | Lowercase Letters | Upper and Lowercase Letters | Numbers, Upper and Lowercase Letters | Numbers, Upper and Lowercase Letters, Symbols |
|---|---|---|---|---|---|
| 4 | Instantly | Instantly | Instantly | Instantly | Instantly |
| 5 | Instantly | Instantly | Instantly | Instantly | Instantly |
| 6 | Instantly | Instantly | Instantly | 1 sec | 5 secs |
| 7 | Instantly | Instantly | 25 secs | 1 min | 6 mins |
| 8 | Instantly | 5 secs | 22 mins | 1 hour | 8 hours |
| 9 | Instantly | 2 mins | 19 hours | 3 days | 3 weeks |
| 10 | Instantly | 58 mins | 1 month | 7 months | 5 years |
| 11 | 2 secs | 1 day | 5 years | 41 years | 400 years |
| 12 | 25 secs | 3 weeks | 300 years | 2k years | 34k years |
| 13 | 4 mins | 1 year | 16k years | 100k years | 2m years |
| 14 | 41 mins | 51 years | 800k years | 9m years | 200m years |
| 15 | 6 hours | 1k years | 43m years | 600m years | 15 bn years |
| 16 | 2 days | 34k years | 2bn years | 37bn years | 1tn years |
| 17 | 4 weeks | 800k years | 100bn years | 2tn years | 93tn years |
| 18 | 9 months | 23m years | 6tn years | 100 tn years | 7qd years |

*Figura 1: Tiempo que lleva averiguar una contraseña. (Hive Systems, 2024)*

## 1.3 Contexto y justificación

La motivación de este proyecto viene dada por el contexto actual, en el que como se ha explicado anteriormente, estamos ante dos industrias que están en pleno crecimiento y desarrollo, con una utilidad inmensa y de las cuales el mercado todavía no ha sabido unir bien para crear un producto de valor. Se está hablando de la industria de la gestión de contraseñas y de la blockchain, donde ambas fusionadas pueden dar un producto que no se haya visto todavía desarrollada de la manera idónea.

Por otro lado, el aumento de concienciación de la privacidad y seguridad en internet por parte de los usuarios es un aspecto clave en el que centrarse para realizar este tipo de proyecto, debido a que multitud de personas ya no se sienten seguras al proporcionar datos personales de alto valor a empresas las cuales no se sabe exactamente qué hacen con dicha información.

En línea con lo anterior, el proyecto que se plantea realizar, aportará al campo de la seguridad de las contraseñas una opción distinta a las actuales, donde se destaca la privacidad al disponer de la tecnología blockchain siendo transparente con los datos del usuario y la seguridad, brindando una descentralización y cifrado de las contraseñas almacenadas.

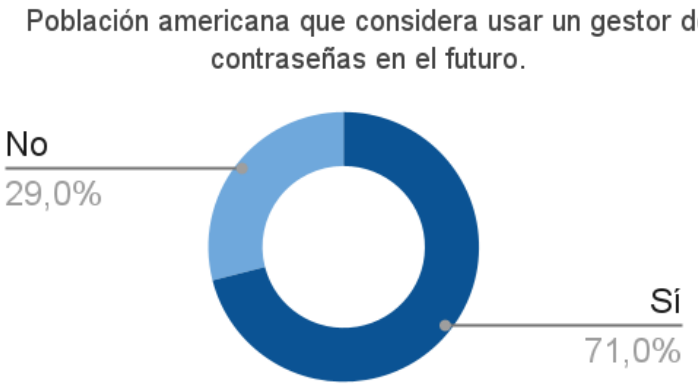


*Figura 2: Población americana que considera usar gestores de contraseñas en un futuro*

## 1.4 Planteamiento del problema

No son aisladas las nuevas aplicaciones que están surgiendo de empresas con mucho renombre, como se ha comentado en el apartado del estado del arte.

Sin embargo, muchos usuarios aún reniegan del uso de estas plataformas, principalmente por falta de confianza para almacenar algo tan importante como lo es una llave que desbloquea toda la información personal de un individuo.

La gran mayoría de los gestores existentes se basan en una tecnología centralizada y administrada por ellos mismos, en lo cual viene implícito dos puntos negativos; al depender de una única base de información esta puede ser vulnerada o mutada por terceros haciendo que los datos del usuario se puedan ver afectados. Por otro lado, si el usuario quiere comprobar cómo y dónde están sus contraseñas almacenadas (teniendo todo su derecho a ello) este no podrá o le será difícil acceder a tal información a causa de la hermeticidad de las empresas dueñas de la plataforma.

En este contexto, la tecnología blockchain surge como una solución para abordar muchos de los desafíos en seguridad y privacidad de la información, incluyendo la gestión de contraseñas, ya que la naturaleza descentralizada y distribuida de la blockchain ofrece un nivel superior de seguridad y transparencia.

Este proyecto pretende enfrentar los desafíos que presenta la gestión de datos personales como lo son las contraseñas, y con ello, que personas que deseen un servicio seguro y transparente de su información, puedan acceder a un sistema para gestionar sus contraseñas en un mismo lugar y a la vez distribuido.

# Capítulo 2. OBJETIVOS

## 2.1 Objetivos generales

El objetivo de este proyecto se basa en diseñar, desarrollar e implementar un sistema de gestión de contraseñas basado en la tecnología blockchain a través de una interfaz web, que permita a los usuarios mediante una clave privada maestra, consultar sus contraseñas.

El sistema garantizará el cifrado de las contraseñas en la blockchain, revelando el contenido legible a aquellos usuarios que posean la clave privada maestra correspondiente.

## 2.2 Objetivos específicos

- Desarrollar una aplicación web “responsive”. La plataforma registrará al usuario y las contraseñas del mismo.
- Registrar e iniciar sesión a usuarios para poder usar las funcionalidades del gestor de contraseñas. Este registro se efectuará sobre una base de datos tradicional NoSQL.
- Crear una clave privada maestra para cifrar las distintas contraseñas.
- Crear, consultar y eliminar las contraseñas del usuario en la blockchain.
- Generar claves aleatorias seguras para que el usuario las pueda utilizar como contraseñas.
- Visualizar cómo y dónde están almacenadas las contraseñas del propio usuario.

## 2.3 Beneficios del proyecto

Los beneficios que aporta el proyecto en relación a los objetivos, es la posesión de una plataforma web intuitiva con la utilización de la tecnología blockchain para almacenar las contraseñas y la visualización de estas, teniendo la opción de saber cómo y dónde se está guardando dicha información personal del usuario.

Debido a la naturaleza de las aplicaciones web, se puede acceder a todas las funcionalidades mencionadas desde cualquier parte del planeta con una conexión a internet.

# Capítulo 3. DESARROLLO DEL PROYECTO

## 3.1 Planificación del proyecto

La planificación del proyecto se ha realizado en base a lo estipulado en el anteproyecto, donde se plantean los siguientes apartados; formación, análisis, diseño, desarrollo, pruebas y documentación. El tiempo dedicado al proyecto se ha distribuido en cuatro fases que están formadas por las tareas que se han llevado a cabo, a continuación se explican más detalladamente.

### 3.1.1 Formación, análisis e inicio del proyecto

Esta fase comienza con la definición de los temas que se van a tratar, con el objetivo de adquirir conocimientos de la industria y el contexto donde esta se encuentra, para más adelante, saber si el proyecto es viable o no. Es un apartado el cual no requiere demasiado esfuerzo, pero si que es muy importante, ya que si este se realiza mal, el trabajo que se desarrolla posteriormente puede estar mal encaminado. Tiene una duración de tres meses, siendo la fase más duradera.

La investigación y formación es la primera tarea en llevarse a cabo, ocupando un tiempo considerable debido a la realización de cursos de blockchain y lecturas relacionadas.

La siguiente tarea es la definición del alcance y los objetivos que iba a contener el proyecto, los cuales se han mencionado en apartados anteriores.

Más tarde, se desarrolla un análisis de requisitos donde se recopila información de funcionalidades, características y restricciones del sistema, para así, cumplir con los objetivos y expectativas del proyecto. Y con esta última información, se realiza el anteproyecto.

### 3.1.2 Definición y planificación del proyecto

Esta fase del proyecto es una de las más notables, debido a que se empieza a definir cómo va a ser el producto final técnicamente en cuanto al almacenamiento de información, el cifrado y la interfaz de usuario. Tiene una duración de un mes, siendo junto a la última fase, la más corta de todas.

Lo primero en esta fase fue diseñar la blockchain, es decir, las funciones, variables, lenguaje, librerías y todo lo relacionado para plantear cómo se desarrollará este en la siguiente fase. Este diseño tuvo un esfuerzo extra, debido al caso peculiar del proyecto en el que había que pensar desde cero cómo se iba a diseñar una blockchain para un uso que no es común, y por lo tanto, no está documentado en la red. En el desarrollo de esta definición, se utilizaron esquemas visuales para así, interpretar correctamente su funcionamiento.

La siguiente tarea consta de más simplicidad que la anterior, y tuvo que ver con la elección de la base de datos y los datos que se van a guardar dentro de esta. Se escogió una NoSQL por rapidez y simplicidad a la hora de implementarlo en una plataforma web, y los datos que se guardan son para la autenticación del usuario y otros datos del mismo.

Una de las tareas determinantes para el funcionamiento correcto de la aplicación es la definición de las reglas de cifrado y descifrado, que como se puede intuir, se trata de establecer las pautas de cómo se va a almacenar la información sensible en la blockchain.

La última tarea de esta fase es el diseño de la interfaz del usuario, siendo la más duradera debido a que uno de los objetivos de la aplicación es tener una interfaz original e intuitiva.

### 3.1.3 Desarrollo del proyecto

Esta fase dura dos meses, y como bien indica su nombre, se empieza a realizar el código de la plataforma web, su backend y la integración entre ambos.

Para el desarrollo de la blockchain, se siguió el diseño hecho con anterioridad, aunque no fue hasta el final de la integración entre el frontend y el backend, cuando se acabó por completo, ya que se tuvieron que añadir pequeños detalles posteriormente.

Posteriormente, se configura la base de datos NoSQL para manejar y crear los campos definidos en tareas anteriores. Esta tarea resultó ser más compleja debido a la limitada documentación oficial disponible sobre la integración con la base de datos seleccionada.

El desarrollo del frontend es la siguiente tarea, donde se realiza el código de toda la interfaz web, teniendo el mayor número de horas dedicadas a una actividad, ya que es la cara visible de la aplicación y debe tener la máxima calidad posible.

La última tarea de esta fase es la integración de todo lo desarrollado en las tres tareas anteriores, unificando todo en una sola plataforma web.

### 3.1.4 Rendimiento y supervisión del proyecto

En este momento, la aplicación ya está terminada y se dispone a realizar las pruebas de su correcto funcionamiento y la documentación. Tiene un mes de duración con una complejidad baja en comparación con fases anteriores.

Las pruebas se realizaron mediante un plan, donde se exponen los usos que debe tener la plataforma y se intenta recrearlos de todas las maneras posibles, en el caso de que ocurra algún error, se corrige.

Por último, es la documentación y después la presentación, unas tareas que no son complejas pero se le debe dedicar el tiempo necesario para que expongan de manera clara y ordenada el trabajo realizado durante los anteriores seis meses.

### 3.1.5 Variaciones en la planificación

El seguimiento de la planificación del proyecto se ha tomado seriamente. Sin embargo, las fases iniciales de la planificación pueden contener errores en el cálculo del tiempo necesario para completar las diferentes tareas. Esto se debe a que se realizan predicciones sobre el esfuerzo y tiempo requeridos, sin poder saberlo con certeza.

El proyecto realizado tiene aproximadamente un tiempo de dedicación de 341 horas, frente a las 308 horas estimadas en el anteproyecto. Este tiempo añadido se ha destinado a mejorar o agregar las siguientes tareas:

- **Análisis de requisitos:** Aunque no se incluyó desde el principio, esta tarea es indispensable para obtener una visión clara de las funcionalidades del sistema.
- **Definición de encriptado y desencriptado:** Esta tarea define, mediante esquemas, los distintos algoritmos utilizados. Se añadió debido a la complejidad que implica el tratamiento de las contraseñas de los usuarios.
- **Diseño de la interfaz gráfica (UX) y desarrollo del frontend:** Se dedicó más tiempo del planificado a estas tareas, debido a necesidades de usabilidad imprevistas, siendo necesario que la plataforma mantuviera un diseño original y atractivo.
- **Documentación:** Se destinó más tiempo a la documentación del proyecto para mejorar su calidad.

A continuación se puede observar la comparativa de los diagramas de Gantt, entre la planificación del anteproyecto y la que finalmente se ha realizado.

***Diagrama de Gantt del anteproyecto***

| TÍTULO DE LA TAREA | FECHA DE INICIO | FECHA DE ENTREGA | HORAS |
|---|---|---|---|
| **Formación, análisis e inicio del proyecto** | | | |
| Formación en Blockchian | 12/11/23 | 22/12/23 | 33 |
| Análisis de viabilidad de proyecto | 01/01/24 | 20/01/24 | 10 |
| Realización de anteproyecto | 20/01/24 | 02/02/24 | 5 |
| **Definición y planificación del proyecto** | | | |
| Definir diseño de blockchain | 03/02/24 | 13/02/24 | 10 |
| Definir datos y elección de base de datos | 14/02/24 | 21/02/24 | 5 |
| Diseñar UX | 22/02/24 | 27/02/24 | 10 |
| **Desarrollo proyecto** | | | |
| Desarrollo de la blockchain | 28/02/24 | 01/04/24 | 55 |
| Desarrollo del backend | 10/03/24 | 06/04/24 | 65 |
| Desarrollo del frontend | 15/03/24 | 17/04/24 | 65 |
| Integración de las diferentes partes | 18/04/24 | 30/04/24 | 20 |
| **Rendimiento/supervisión del proyecto** | | | |
| Pruebas y depuración | 01/05/24 | 04/05/24 | 5 |
| Documentación | 05/05/24 | 17/05/24 | 20 |
| Presentación | 18/05/24 | 31/05/24 | 5 |
| | | | 308 |


*Figura 3: Diagrama de Gantt del anteproyecto*

***Diagrama de Gantt finalmente seguido***

| TÍTULO DE LA TAREA | FECHA DE INICIO | FECHA DE ENTREGA | HORAS |
|---|---|---|---|
| **Formación, análisis e inicio del proyecto** | | | |
| Investigación y formación preliminar | 12/11/23 | 22/12/23 | 33 |
| Definición del alcance y objetivos | 01/01/24 | 19/01/24 | 3 |
| Anteproyecto | 20/01/24 | 02/02/24 | 4 |
| Análisis de requisitos | 20/01/24 | 03/02/24 | 10 |
| **Definición y planificación del proyecto** | | | |
| Definir diseño de blockchain | 03/02/24 | 13/02/24 | 5 |
| Definir datos y elección de base de datos | 14/02/24 | 21/02/24 | 5 |
| Definir encriptado y desencriptado | 16/02/24 | 21/02/24 | 10 |
| Diseñar UX | 22/02/24 | 03/03/24 | 15 |
| **Desarrollo del proyecto** | | | |
| Desarrollo de la blockchain | 13/03/24 | 13/04/24 | 60 |
| Desarrollo de la base de datos | 10/03/24 | 15/04/24 | 50 |
| Desarrollo del frontend | 27/03/24 | 17/04/24 | 65 |
| Integración de las diferentes partes | 18/04/24 | 30/04/24 | 40 |
| **Rendimiento/supervisión del proyecto** | | | |
| Pruebas y depuración | 01/05/24 | 04/05/24 | 5 |
| Documentación | 05/05/24 | 17/05/24 | 31 |
| Presentación | 18/05/24 | 31/05/24 | 5 |
| | | | 341 |

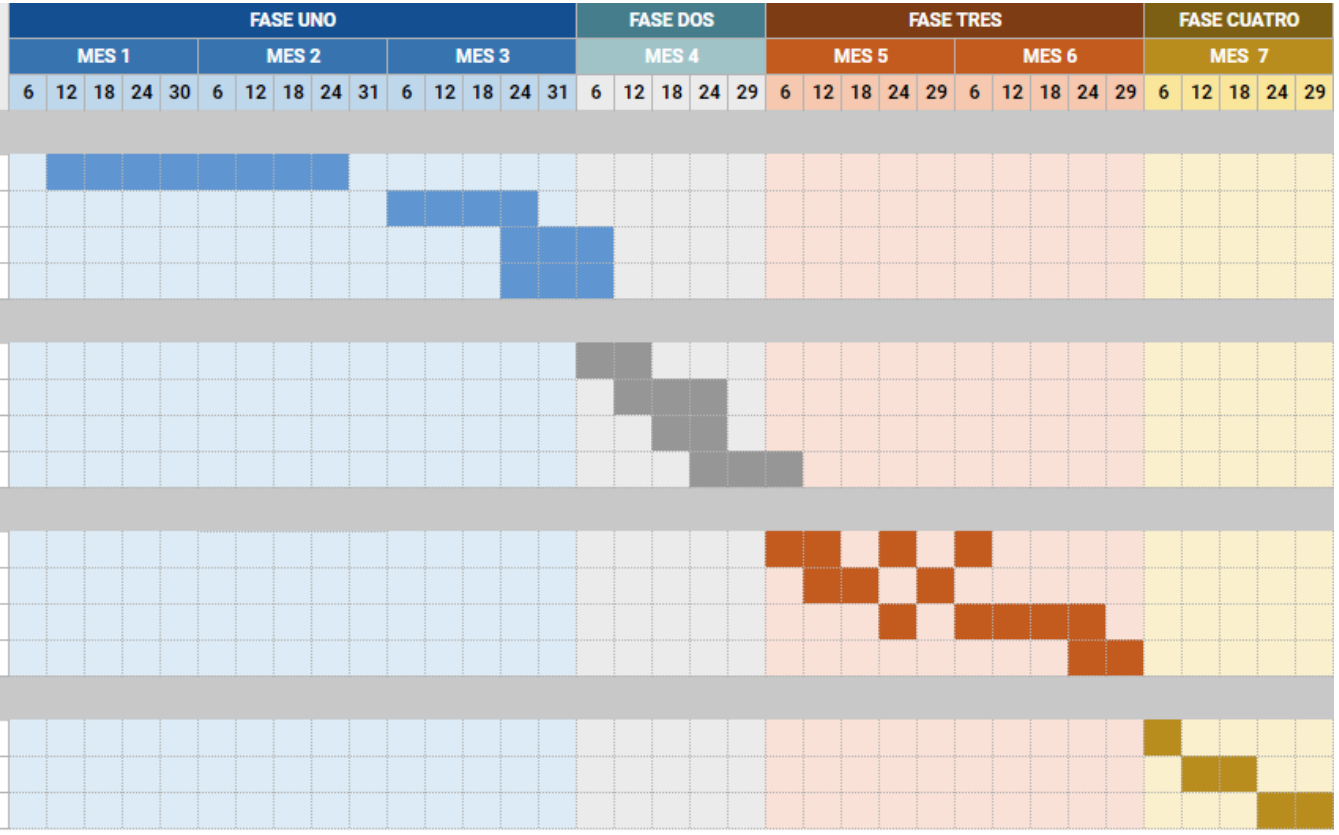


*Figura 4: Diagrama de Gantt finalmente seguido*

## 3.2 Descripción de la solución, metodologías y herramientas empleadas

### 3.2.1 Arquitectura propuesta

La arquitectura hardware general del proyecto se basa en un modelo cliente-servidor. Es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta.

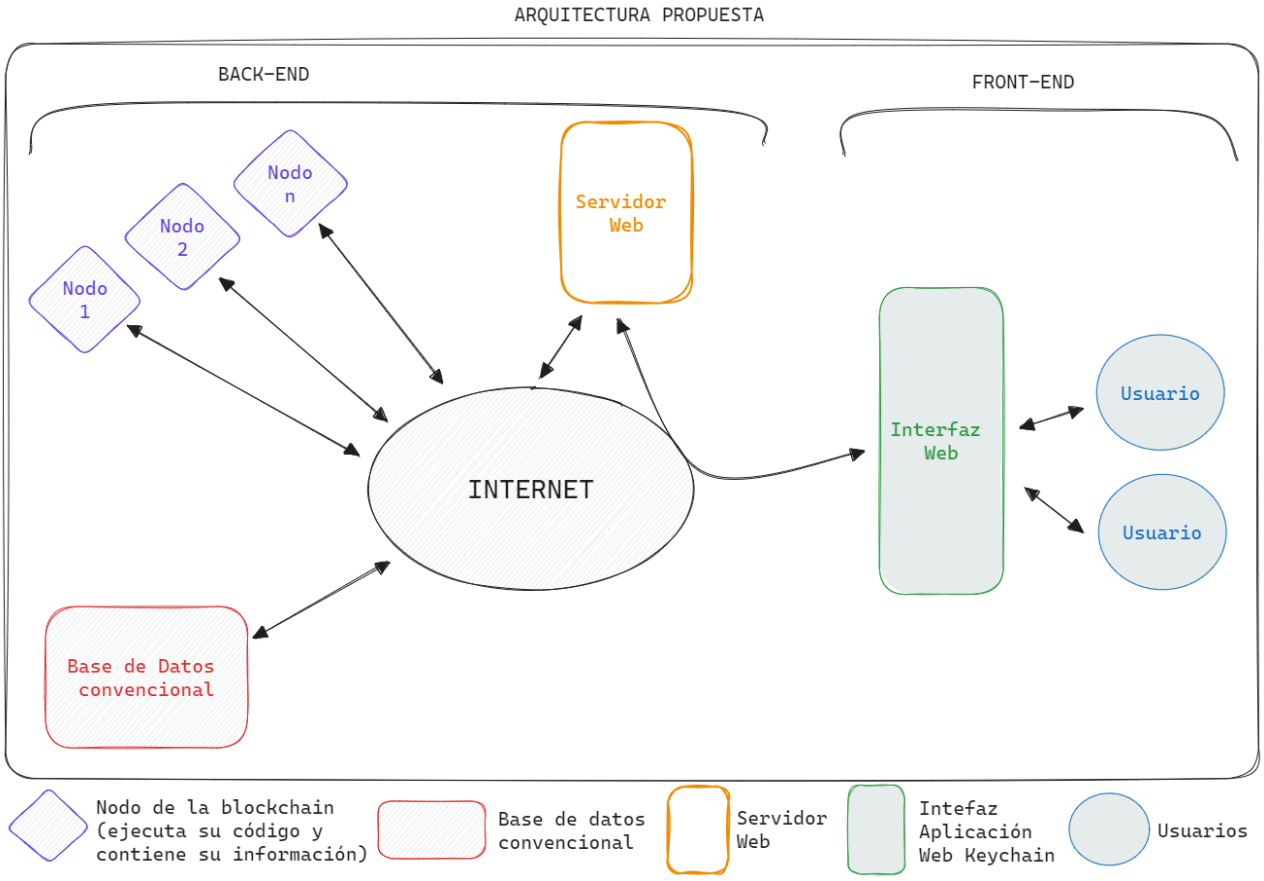


*Figura 5: Arquitectura general de la aplicación*

Por otro lado, la arquitectura se integra con la red de la blockchain, la cual está compuesta por distintos nodos (servidores que ejecutan una copia idéntica del código e información de la blockchain) repartidos por distintos lugares en el mundo, siendo así, una red descentralizada.

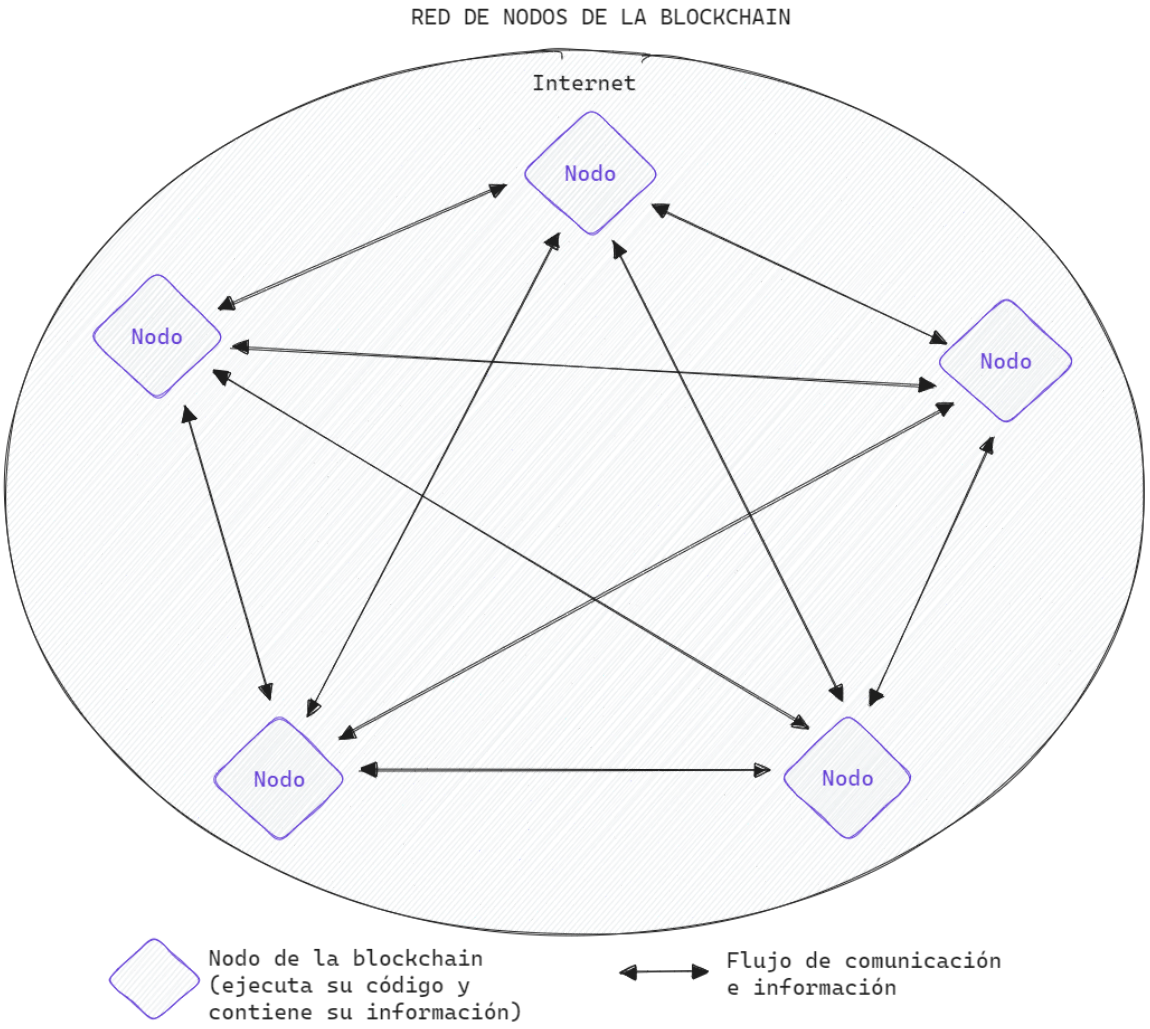


*Figura 6: Red de los nodos de la blockchain*

***Cliente/Usuario***

- Los clientes son los dispositivos que acceden a la aplicación web, como portátiles, ordenadores de escritorio, tabletas o teléfonos inteligentes.
- Estos dispositivos ejecutan el frontend de la aplicación y este es responsable de interactuar con los usuarios, presentar la interfaz de usuario y enviar solicitudes al servidor.
- Los clientes también manejan la generación de la clave privada maestra y el cifrado de las contraseñas antes de enviarlas al servidor para su almacenamiento en la blockchain.

***Servidor***

- El servidor es el componente central del sistema que maneja la lógica de negocio, el procesamiento de las solicitudes de los clientes y la interacción con la blockchain.
- Está compuesto por la aplicación web y el backend, desarrollado en Python con Flask en este caso, que proporciona API endpoints para que el frontend se comunique con él.
- El servidor también interactúa con la base de datos, que incluye Firebase para la autenticación y para almacenar cierta información del usuario.
- Además, el servidor ejecuta nodos de la blockchain privada, que almacena y procesa los bloques de información relacionados con las contraseñas.

***Blockchain***

- La blockchain es una red descentralizada de nodos que almacenan una copia idéntica del "libro de contraseñas".
- En este caso, se implementa una blockchain privada para garantizar la privacidad y la confidencialidad de las contraseñas.
- Cada nodo de la blockchain almacena los bloques de información que utiliza la aplicación para guardar correctamente las contraseñas.
- La blockchain está formada por los bloques enlazados entre sí criptográficamente usando funciones hash, formando como su propio nombre indica, una cadena de bloques.

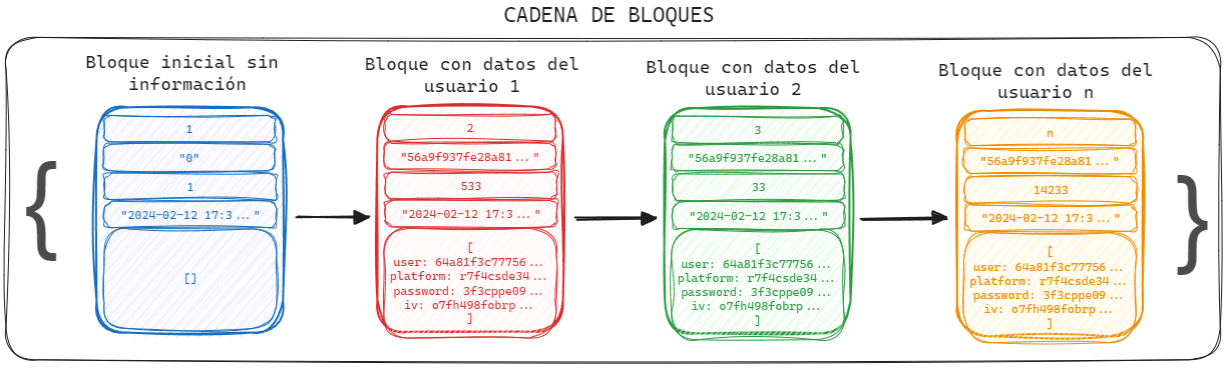


*Figura 7: Cadena de bloques*

***Base de datos***

- La base de datos elegida es de tipo NoSQL documental.
- Es encargada de la autenticación de la aplicación.
- Almacena ciertos datos del usuario.

***Comunicación***

- La comunicación entre el cliente y el servidor, así como entre los nodos de la blockchain, se realiza a través de Internet utilizando protocolos seguros estándar como HTTP(S) sobre certificados SSL.

La arquitectura hardware detrás de este sistema cliente-servidor incluye dispositivos clientes que ejecutan el frontend de la aplicación web, un servidor central que maneja la lógica de negocio y la interacción con la blockchain, y nodos de blockchain que almacenan y procesan los datos de las contraseñas de forma descentralizada.

### 3.2.2 Tecnologías, dispositivos y herramientas empleadas

En el desarrollo de este proyecto se han empleado una serie de tecnologías y herramientas que han permitido crear una aplicación web robusta y segura para la gestión de contraseñas. Desde el frontend hasta el backend y la gestión de datos, cada componente ha sido seleccionado para garantizar la eficiencia y la seguridad del sistema.

***Python***

Python es un lenguaje de programación de alto nivel, interpretado y de propósito general. Es conocido por su sintaxis simple y legible, pero también poderoso y versátil para proyectos avanzados.

El principal motivo para su uso es la extensa documentación que existe del lenguaje en la red, haciendo que los problemas que ocurran, ya estén identificados en internet.

También ofrece multitud de recursos para usarlo como un lenguaje de back-end, ya que en los últimos años se ha utilizado para esta labor más frecuentemente.

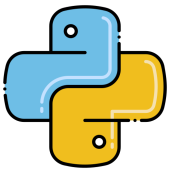

*Figura 8: Logo Python*

***JavaScript***

JavaScript es un lenguaje de programación flexible que puede adaptarse a diferentes requisitos de proyecto y escalar según sea necesario.

Principalmente se utiliza para realizar aplicaciones web interactivas, y una de las ventajas de usar este lenguaje, al igual que con Python, es lo bien documentado que se encuentra en internet, teniendo una adopción masiva en todo el mundo.


*Figura 9: Logo JavaScript*

***JSON***

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos. Se utiliza comúnmente para representar datos estructurados de manera legible para humanos y fácilmente procesables por las máquinas. Está basado en una sintaxis de objetos y listas que se asemeja a la notación de objetos en JavaScript.

Este formato se utiliza en el proyecto para guardar los datos de la blockchain, es decir, los bloques de la blockchain se representan en objetos JSON, para así, poder visualizarlos y procesarlos adecuadamente.

***HTML***

HTML (HyperText Markup Language) es el lenguaje estándar utilizado para crear y diseñar páginas web. Se compone de una serie de elementos o etiquetas que definen la estructura y el contenido de una página web.

En el proyecto, se ha utilizado para realizar la página web, sin embargo, al usar la biblioteca React, el desarrollo en este lenguaje pasa a tener un papel secundario.

***CSS***

CSS (Cascading Style Sheets) es un lenguaje de estilo utilizado para describir la presentación de documentos HTML y XML. Permite definir cómo se deben mostrar los elementos HTML en términos de diseño, formato, colores y otros aspectos visuales.

En la aplicación desarrollada se ha utilizado para dotarla de estilos y diseño, y con ello tener como resultado una página web con una interfaz gráfica original y fácil de usar.

***React***

React es una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas y reutilizables. Desarrollada por Facebook, React permite a los desarrolladores crear componentes de interfaz de usuario que gestionan su propio estado y se actualizan de manera eficiente cuando cambian los datos.

React se programa mediante componentes, los cuales son piezas de la interfaz de usuario que puede contener HTML, CSS y JavaScript. En ella se utiliza JSX, una extensión de JS y HTML.

Se ha utilizado esta tecnología para realizar el frontend de la aplicación web, y debido a que uno de los objetivos del proyecto es tener una interfaz de usuario original e intuitiva, es una de las mejores opciones que existen.

*Figura 10: Logo Reat*

***CryptoJS***

CryptoJS es una biblioteca de JavaScript que proporciona funciones para realizar operaciones criptográficas en el navegador web. Permite a los desarrolladores cifrar y descifrar datos utilizando algoritmos de cifrado como AES, así como funciones hash como SHA-256.

En el proyecto, esta biblioteca se ha usado para el tratamiento de todos los datos sensibles que la aplicación necesita ocultar para que la información no sea legible por usuarios terceros no autorizados. Los algoritmos utilizados de esta biblioteca han sido AES para el cifrado simétrico, SHA-3 como función hash y PBKDF2 como función de derivación de clave.

***Flask***

Flask es un microframework web ligero y flexible para Python. Es conocido por su simplicidad y facilidad de uso, lo que lo hace ideal para construir aplicaciones web pequeñas y medianas de manera rápida y eficiente.

Flask es la herramienta que se ha elegido para realizar la API, la cual es la encargada de comunicarse con la blockchain mediante las peticiones que el usuario quiera realizar.

***Otras bibliotecas y módulos***

**react-router-dom:** React Router es una biblioteca de enrutamiento para React que permite a los desarrolladores agregar enrutamiento a sus aplicaciones web de React.

**Hashlib**: Es una biblioteca de Python que proporciona funciones para calcular hashes de datos.

**Requests**: Es una biblioteca de Python utilizada para enviar solicitudes HTTP/1.1.

**Urlparse**: Es un módulo de Python que proporciona funciones para analizar y manipular URLs.

**Datetime**: Es un módulo de Python que proporciona clases para trabajar con fechas y horas.

**Os**: Es un módulo de Python que proporciona funciones para interactuar con el sistema operativo subyacente.

**CORS**: CORS significa Cross-Origin Resource Sharing y es una técnica utilizada en aplicaciones web para permitir que los recursos de un sitio web se compartan con otro dominio diferente.

**Uuid4**: Es una función en el módulo uuid de Python que se utiliza para generar identificadores únicos universalmente (UUID).

***Base de datos Firebase***

Firebase es una plataforma de desarrollo de aplicaciones móviles y web desarrollada por Google. Proporciona una amplia gama de servicios y herramientas que facilitan la creación, el desarrollo y la administración de aplicaciones.

Firebase se ha utilizado en el proyecto para dos funcionalidades; la autenticación mediante correo electrónico y contraseña, y para almacenar datos del usuario.

En el caso de la autenticación, Firebase Authentication proporciona servicios con facilidad de integración en la aplicación web, con ello sumado a que el plan de pago comienza según la popularidad del sistema, es decir, si la base de datos no consume unas peticiones determinadas, su coste es cero.

En el caso de la base de datos más tradicional, se utiliza Firestore, una base de datos NoSQL documental la cual es una opción idónea para la aplicación web debido a la facilidad de integración con el backend y la rapidez de las respuestas que esta herramienta ofrece.

*Figura 11: Logo Firebase*

***GitHub***

GitHub es una plataforma de desarrollo colaborativo basada en la nube que permite a los desarrolladores alojar, revisar y colaborar en proyectos de software utilizando el sistema de control de versiones Git.

Para este proyecto se creó un repositorio privado de Github con el objetivo de guardar el código de la aplicación web y poder acceder a él desde otro dispositivo de forma privada.

*Figura 12: Logo Github*

***Visual Studio Code***

Visual Studio Code es un editor de código fuente desarrollado por Microsoft que se ha vuelto extremadamente popular entre los desarrolladores. Es conocido por su velocidad, flexibilidad y extensibilidad.

Con este editor de código se ha desarrollado únicamente la parte de la aplicación web del frontend, ya que para el desarrollo web es el más recomendable.

*Figura 13: Logo Visual Studio Code*

***PyCharm***

PyCharm es un entorno de desarrollo integrado (IDE) específicamente diseñado para el desarrollo de aplicaciones Python. Es un editor muy intuitivo y profesional, sin embargo, la versión con mayor calidad requiere una licencia que se adquiere con un plan de pago o bien con licencia de estudiante.

Este IDE se ha utilizado para el desarrollo de la blockchain y la API que interactúa con los nodos de la red.

*Figura 14: Logo PyCharm*

***Postman***

Postman es una herramienta de desarrollo de API que facilita la creación, prueba, documentación y colaboración en torno a APIs. Permite enviar solicitudes HTTP a un servidor y analizar las respuestas.

Este programa de escritorio se ha usado para comprobar y desarrollar la API que interactúa con la blockchain, con el fin de realizar depuración y pruebas en un entorno controlado.

*Figura 15: Logo Postman*

***Notion***

Notion es una herramienta todo en uno que combina la funcionalidad de varias aplicaciones de productividad en una sola plataforma. Es conocida por su flexibilidad y capacidad para adaptarse a una amplia variedad de casos de uso.

En este proyecto Notion se ha usado para gestionar las tareas y documentar parte de estas, como por ejemplo indicar notas rápidas para más adelante realizar la memoria de la aplicación, también se ha organizado los intervalos de tiempo que requieren todas las tareas para así tener una vista general de cómo se va desarrollando los tiempos del proyecto.

Otro uso que se le ha dado a esta plataforma es la creación de la política de privacidad utilizada por la aplicación, donde gracias a poder compartir páginas con otros usuarios, se pueden crear notas con el acceso a ellas desde solamente un link.

***Canva***

Canva es una plataforma en línea que permite a los usuarios crear una variedad de diseños gráficos de manera fácil y rápida, incluso sin experiencia previa en diseño.

El uso de Canva en este proyecto ha sido para la creación de bocetos y diseños de la aplicación web, incluyendo todas las pantallas de la web, logos, iconos, paleta de colores y todo lo relacionado con el diseño gráfico.

***Excalidraw***

Excalidraw es una herramienta de dibujo en línea que permite a los usuarios crear diagramas y bocetos de forma colaborativa y en tiempo real.

La utilización de esta plataforma web ha servido para realizar todos los diagramas del proyecto con el fin de poder visualizar y plantear de manera sencilla todo el desarrollo de la aplicación, desde las tecnologías que se iban a usar, hasta el flujo de los datos de un determinado algoritmo.

***OneDrive***

OneDrive es un servicio de almacenamiento en la nube desarrollado por Microsoft que permite a los usuarios almacenar, sincronizar y compartir archivos y datos en línea.

Esta herramienta se ha utilizado para compartir automáticamente entre distintos dispositivos recursos necesarios para la realización del proyecto, como por ejemplo el archivo python con el código de la blockchain y la API, o el archivo de Excalidraw anteriormente mencionado con todos los diagramas.

***Equipos con Windows 11***

Para la realización del proyecto, se han usado ordenadores con el sistema operativo Windows 11 para todas las tareas, especialmente para el desarrollo del software. Además, con estos equipos se han utilizado las tecnologías y herramientas anteriores.

Se contaba con un ordenador de sobremesa para avanzar en el desarrollo del proyecto desde el domicilio personal, y con un portátil, que se utiliza para trabajar desde cualquier otro lugar.

### 3.2.3 Metodologías

En el contexto del desarrollo de software, las metodologías juegan un papel fundamental al proporcionar un marco de trabajo estructurado y sistemático para la planificación, ejecución y entrega de proyectos. Estas metodologías son conjuntos de prácticas, procesos y principios que guían el desarrollo de software desde la concepción hasta la implementación.

Para la realización de este proyecto se ha seleccionado seguir una metodología en cascada, que debido a su naturaleza estructurada y secuencial, se alinea bien con la naturaleza definida y predecible de los requisitos y procesos involucrados en el desarrollo de un gestor de contraseñas basado en la blockchain.

Dado que el proyecto tiene una definición clara de los objetivos, requisitos y entregables desde el principio, la metodología en cascada permite un enfoque paso a paso, donde cada etapa se completa antes de pasar a la siguiente.

La metodología de cascada generalmente tiene las etapas de requerimientos, diseño, implementación, verificación y mantenimiento.

En el caso de este proyecto, las fases y tareas que se han realizado engloban el mismo contenido que las etapas mencionadas, teniendo las siguientes:

- Formación, análisis e inicio del proyecto: donde se especifican los requerimientos.
- Definición y planificación del proyecto: donde se define el diseño que tendrá la aplicación.
- Desarrollo del proyecto: es donde se implementa el código y se realiza la aplicación.
- Supervisión del proyecto: esta fase engloba la verificación del correcto comportamiento de la aplicación y su mantenimiento posteriormente.


*Figura 16: Planificación seguida en la aplicación Notion*

## 3.3 Recursos requeridos

***Lenguajes de programación***

- Python
- JavaScript
- HTML
- CSS
- JSON

***Bibliotecas***

- React
- react-router-dom
- crypto-js
- Flask
- HashLib
- request
- urlparse
- datetime
- os
- CORS
- uuid4

***Programas***

- Visual Studio Code
- PyCharm
- Postman

***Plataformas online***

- Github
- Firebase
- Notion
- Canva
- Excalidraw
- OneDrive

***Dispositivos***

- Equipos con Windows 11

***Recursos técnicos***

- Curso de 27 horas “Smart Contracts y Blockchain con Solidity de la A a la Z”, de la academia Udemy.
- Lectura “Bitcoin Con Rigor” de Jose Sanchis en colaboración con Racks Labs.
- Lectura de documentación oficial de Ethereum.

## 3.4 Presupuesto

El presupuesto de un proyecto es una herramienta fundamental que permite estimar y planificar los recursos financieros necesarios para llevar a cabo todas las actividades y objetivos propuestos. Constituye una guía financiera que ayuda a gestionar los costos, controlar los gastos y garantizar la viabilidad económica del proyecto en su conjunto.

En esta sección, se detallan los costos asociados a cada componente del proyecto, incluyendo el software, infraestructura tecnológica, mano de obra, entre otros.

| Nombre de fases | Proyecto (7 MESES) | |
|---|---|---|
| *Categorías* | *Descripción y unidades* | *Costes/proyecto* |
| Mano de obra | | 7.752 € |
| Alfonso Vilchez de las Heras (director del proyecto) | 12 horas trabajadas. 35,0 | 420 € |
| Yago Iglesias Díaz (arquitecto de software y desarrollador) | 341 horas trabajadas. 21,5€ la hora | 7.332 € |
| Hardware | | 2.158 € |
| Ordenador de sobremesa (servidor web) | 1 unidad, ordenador con procesador Ryzen 5700X, Tarjeta gráfica AMD RX 7600 XT, placa base ASUS B550-PUS, Ram 32 Gb, Almacenamiento 1TB, Refrigeración por aire, Fuente de 750W modular | 1.100 € |
| Portátil | 1 unidad, portatil Xiaomi Mi Air 13.3" | 898 € |
| Monitor | 2 unidades, monitores IPS Full HD | 160 € |
| Software | | 343 € |
| IDE PyCharm | 1 licencia, 28.9 € al mes | 202 € |
| IDE Visual Studio Code | 1 unidad | 0 € |
| Postman | 1 unidad | 0 € |
| Windows 11 | 1 unidad, versión Pro | 29 € |
| Github | 1 licencia, 4 € al mes | 28 € |
| Firebase | Número limitado de peticiones sin conste | 0 € |
| Notion | 1 licencia | 0 € |
| Canva | 1 licencia, 12 € al mes | 84 € |
| Excalidraw | 1 licencia | 0 € |
| Onedrive | Hasta 5Gb sin coste | 0 € |
| Lenguajes de programación | Python, JavaScript, HTML, CSS, JSON | 0 € |
| Bibliotecas | React, react-router-dom, crypto-js, Flask, HashLib, request, urlparse, datetime, os, CORS, uuid4 | 0 € |
| Formacion | | 112 € |
| *Curso online* | 1 unidad de curso Curso "Smart Contracts y Blockchain con Solidity de la A a la Z", de la academia Udemy. | 85 € |
| *Libro* | 1 unidad libro tapa blanda "Bitcoin Con Rigor" de Jose Sanchis en colaboración con Racks Labs. | 27 € |
| *Documentaicón Ethereum* | Documentación online | 0 € |
| **TOTAL** | | **10.364,50 €** |

*Tabla 1: Presupuesto económico total del proyecto desglosado*

## 3.5 Disposiciones legales

La aplicación propuesta se ha desarrollado teniendo en cuenta las distintas limitaciones legales y el tratamiento de los datos que estas leyes dictan. Las disposiciones legales que afectan directamente a la aplicación son las siguientes.

***LOPD y GDPR***

En el ámbito español, la Ley Orgánica de Protección de Datos de Carácter Personal, y en el ámbito europeo, Reglamento General de Protección de Datos. La plataforma desarrollada debe cumplir con la ley y reglamento mencionados, que garantizan los derechos de los ciudadanos ante el deber constitucional de respetar la intimidad de los datos, es decir, la aplicación deberá respetar mediante una serie de normas pautadas el tratamiento de los datos de los usuarios. Estos derechos son:

- Derecho de Acceso: Los usuarios tienen derecho a saber qué datos personales se están recopilando y cómo se están utilizando. Deben poder solicitar y recibir una copia de estos datos.
- Derecho de Rectificación: Los usuarios pueden corregir datos personales inexactos o incompletos.
- Derecho de Supresión: Los usuarios pueden solicitar la eliminación de sus datos personales.
- Derecho a la Limitación del Tratamiento: Los usuarios pueden solicitar la limitación del tratamiento de sus datos personales en ciertas circunstancias.
- Derecho a la Portabilidad de los Datos: Los usuarios pueden recibir sus datos en un formato estructurado y comúnmente utilizado, y tienen derecho a transferirlos a otro responsable.
- Derecho de Oposición: Los usuarios pueden oponerse al tratamiento de sus datos personales en determinadas situaciones.

(Boletín Oficial del Estado [BOE], s.f.)

La aplicación mediante la política de privacidad que proporciona, la cual el usuario deberá de leer, hace posible que los individuos que utilicen la plataforma, dispongan de estos derechos.

Por otro lado, estas reglas también contienen pautas de cómo se deben almacenar los datos personales, en el caso de la aplicación, contraseñas, y estas nunca deben estar guardadas en formato legible y siempre cifradas o hasheadas. En el proyecto se utiliza el algoritmo AES para volver estos datos ilegibles. (Parra, S. 2011)

## 3.6 Análisis de requisitos

### 3.6.1 Requisitos funcionales

Este primer tipo de requisitos detalla las funcionalidades que el sistema debe proporcionar a los usuarios.

| ID + Nombre | Categoría | Descripción | Priorización / Importancia |
|---|---|---|---|
| RQ01 - Registro de usuario | Usuario | Permitir a los usuarios poder registrarse con correo electrónico, nombre de usuario y contraseña en la aplicación web.<br>● RQ01-01: Se requiere rellenar todos los campos | 1(Alto) |
| RQ02 - Inicio de sesión en la aplicación | Usuario | El sistema permitirá a los usuarios poder iniciar sesión con su correo electrónico y contraseña en la aplicación web. | 1(Alto) |
| RQ03 - Creación de clave privada maestra | Usuario | El sistema permitirá a los usuarios crear una clave privada maestra irremplazable para acceder a las contraseñas.<br>● RQ03-01: Se requiere rellenar el campo | 1(Alto) |
| RQ04 - Generación de contraseña aleatoria segura | Usuario | El sistema permitirá a los usuarios generar una contraseña segura aleatoria.<br>● RQ04-01: La contraseña generada se puede copiar mediante un botón | 3(Bajo) |
| RQ05 - Añadir contraseña | Usuario | La aplicación web permitirá añadir contraseña para almacenarla en la blockchain.<br>● RQ05-01: Se requiere haber insertado la clave privada maestra antes.<br>● RQ05-02: Se requiere completar los campos de “plataforma” y “contraseña”. | 1(Alto) |
| RQ06 - Modificar contraseña | Usuario | La aplicación web permite modificar una contraseña específica.<br>● RQ06-01: Se requiere haber insertado la clave privada maestra antes.<br>● RQ06-02: Se requiere editar los | 2(Medio) |

| | | | |
|---|---|---|---|
| | | campos de “plataforma” o “contraseña” | |
| RQ07 - Eliminar contraseña | Usuario | La aplicación web permite eliminar una contraseña específica.<br>● RQ07-01: Se requiere haber insertado la clave privada maestra antes. | 2(Medio) |
| RQ08 - Consultar contraseñas almacenadas | Usuario | El sistema mostrará todas las contraseñas del usuario.<br>● RQ08-01: Se requiere haber insertado la clave privada maestra antes. | 1(Alto) |
| RQ09 - Consultar contraseñas cifradas en la blockchain | Usuario | El sistema permitirá mostrar al usuario todas sus contraseñas cifradas almacenadas en la blockchain.<br>● RQ09-01: Se requiere haber insertado la clave privada maestra antes. | 3(Bajo) |
| RQ010 - Consultar Política de privacidad | Usuario | El sistema permitirá consultar la política de privacidad de la aplicación. | 3(Bajo) |

*Tabla 2: Requisitos funcionales*

### 3.6.2 Requisitos no funcionales

Los requisitos no funcionales detallan las funcionalidades relacionadas con el rendimiento, seguridad y usabilidad.

| ID + Nombre | Categoría | Descripción | Priorización / Importancia |
|---|---|---|---|
| RQNF01 - Rendimiento de la API | Rendimiento de la Aplicación | La API debe tener protocolos en cuanto al rendimiento de la API.<br>● RQNF01-01: Las peticiones GET están controladas por un AbortController, el cual detiene la petición si el usuario la cancela.<br>● RQNF01-02: Las peticiones GET y POST devolverán un mensaje si se ha completado con éxito o no su cometido. | 1(Alto) |

| | | | |
|---|---|---|---|
| | | • RQNF01-02: La API será capaz de gestionar varias peticiones simultáneas. | |
| RQNF02 - Comunicación entre backend y frontend | Comunicación de la Aplicación | El sistema dispone de protocolos para comunicar el frontend con el backend.<br>• RQNF02-01: El formato de intercambio de datos entre el frontend y backend es a través de JSON.<br>• RQNF02-02: La información que se almacena en la blockchain está en formato JSON | 1(Alto) |
| RQNF03 - Interfaz gráfica | Usabilidad de la Aplicación | La interfaz gráfica tiene los siguientes cometidos:<br>• RQNF03-01: La interfaz gráfica debe ser original e intuitiva para la satisfacción de los usuarios.<br>• RQNF03-02: La aplicación web debe tener la cualidad de ser “responsive” y funcionar en los navegadores más demandados.<br>• RQNF03-03: El usuario no debe lidiar con las complejidades de la blockchain. | 2(Medio) |
| RQNF04 - Base de datos | Usabilidad de la Aplicación | La base de datos debe guardar los siguientes datos para el funcionamiento de la aplicación:<br>• RQNF04-01: La base de datos será de tipo NoSQL documental.<br>• RQNF04-02: La base de datos guardará los datos para la autenticacióndel usuario.<br>• RQNF04-03: La base de datos guardará otros datos del usuario para la funcionalidad de gestionar las contraseñas | 1(Alto) |
| RQNF05 - Registro de nombre de usuario | Usabilidad de la Aplicación | Los usuarios deben registrarse con un nombre de usuario único el cual no puede existir en la base de datos. | 1(Alto) |

| | | | |
|---|---|---|---|
| RQNF06 - Limitaciones clave privada maestra | Seguridad de la Aplicación | El sistema limita ciertas características para que la aplicación tenga una mayor seguridad.<br>● RQNF06-01: La clave privada debe tener al menos, 8 caracteres, 1 mayúscula, 1 carácter especial, 1 dígito y 1 minúscula<br>● RQNF06-02: La clave privada maestra sólo se puede registrar una vez y es irremplazable.<br>● RQNF06-03: Sin la clave privada maestra, las contraseñas no se pueden recuperar. | 1(Alto) |
| RQNF07 - Insertar y modificar contraseñas | Seguridad de la Aplicación | Al insertar y modificar las contraseñas, el sistema deberá tener en cuenta los siguientes requisitos de seguridad:<br>● RQNF07-01: Las contraseñas que se inserten deben estar cifradas antes de mandarse al backend y la blockchain.<br>● RQNF07-02: Las contraseñas que se consulten no se descifran en el backend, se envían al frontend cifradas para descifrarlas allí. | 1(Alto) |
| RQNF08 - Datos en la blockchain | Seguridad de la Aplicación | Al almacenar los datos del usuario en la blockchain, el sistema deberá tener en cuenta los siguientes requisitos de seguridad:<br>● RQNF08-01: El campo del UID del usuario está hasheado.<br>● RNFQ08-02: El campo de la plataforma está hasheado.<br>● RQNF08-03: En el caso que se almacena el nombre de usuario, este se almacena hasheado.<br>● RQNF08-04: Los datos restantes se encuentran cifrados. | 1(Alto) |

*Tabla 3: Requisitos no funcionales*

### 3.6.3 Casos de uso

Los casos de uso describen los distintos escenarios específicos de interacción del usuario con el sistema desarrollado.

Las funcionalidades principales de la aplicación son:

- Añadir, modificar y consultar contraseñas.

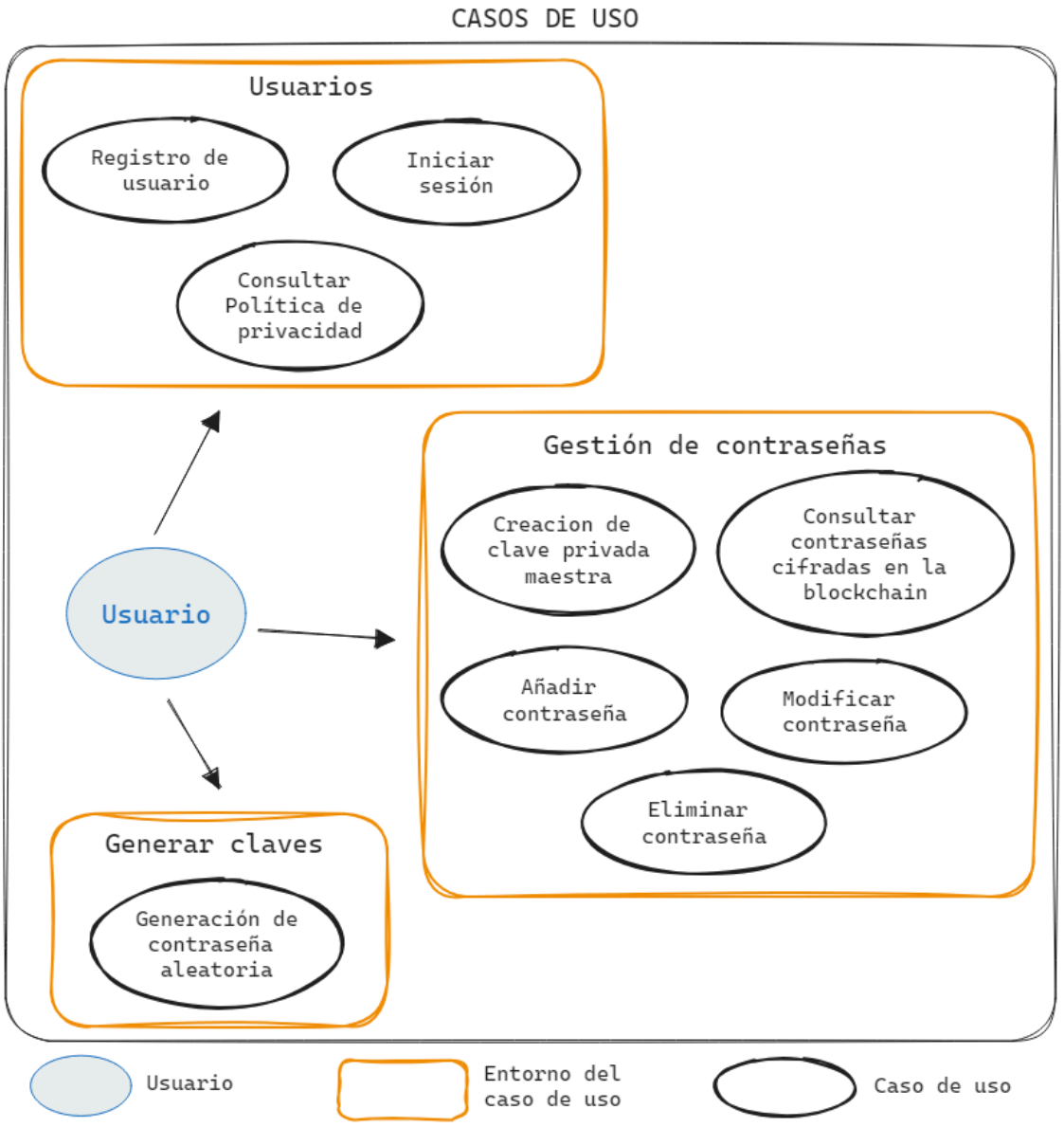


*Figura 17: Casos de uso*

***Registro de usuario***

| | | |
|---|---|---|
| **Caso de Uso CU01** | Registro de usuario | |
| **Objetivo** | Administración de usuarios | |
| **Precondiciones** | Nuevo usuario para registrarse y aplicación web y base de datos operativa | |
| **Postcondiciones** | El usuario queda registrado en la base de datos correctamente y se redirige a la página principal de la web. | |
| **Actores primarios** | Usuario | |
| **Actores secundarios** | Aplicación web, Base de datos Firebase | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación |
| | 2 | Pulsa el botón de iniciar sesión y después de registrar |
| | 3 | Se rellenan todos los campos y se pulsa en aceptar |
| **Extensiones** | **Paso** | **Acción** |
| | 3 | El campo de gmail o username ya existen, se pide que se cambien los campos |

*Tabla 4: Caso de uso: Registro de usuario*

***Inicio de sesión***

| | |
|---|---|
| **Caso de Uso CU02** | Inicio de sesión |
| **Objetivo** | Acceder a la aplicación iniciando sesión. |
| **Precondiciones** | El usuario existe en la base de datos y tener operativas la aplicación web y base de datos. |
| **Postcondiciones** | El usuario inicia sesión correctamente y se redirige hacia la vista principal de la aplicación. |
| **Actores primarios** | Usuario. |
| **Actores secundarios** | Aplicación web, Base de datos Firebase. |

| **Descripción** | **Paso** | **Acción** |
|---|---|---|
| | 1 | El usuario accede a la aplicación. |
| | 2 | Pulsa el botón de iniciar sesión. |
| | 3 | Se rellenan todos los campos y se pulsa en aceptar. |
| **Extensiones** | **Paso** | **Acción** |
| | 3 | El campo de gmail o contraseña son incorrectos, se pide de nuevo los campos. |

*Tabla 5: Caso de uso: Inicio de sesión*

***Consultar la política de privacidad***

| **Caso de Uso CU03** | Consultar la política de privacidad de la aplicación. | |
|---|---|---|
| **Objetivo** | Informar al usuario de la política de privacidad de la aplicación. | |
| **Precondiciones** | Usuario utilizando la plataforma, teniendo operativo la aplicación web y página de Notion de política de privacidad. | |
| **Postcondiciones** | Se redirige al usuario a la página de Notion donde se expone la política de privacidad de la aplicación. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Página de Notion. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación. |
| | 2 | Pulsa el botón de “Política de privacidad” en la página de registro o al final de la página de presentación de la web. |

*Tabla 6: Caso de uso: Consultar la política de privacidad*

***Creación de clave privada maestra***

| | | |
|---|---|---|
| **Caso de Uso CU04** | Creación de clave privada maestra | |
| **Objetivo** | Registrar en la blockchain y base de datos la existencia de la clave privada maestra. | |
| **Precondiciones** | El usuario ya ha iniciado sesión, la aplicación web y base de datos están operativas y no se ha registrado una clave privada maestra antes. | |
| **Postcondiciones** | Se crea un bloque de la blockchain con los datos de autenticación cifrados y se cambia el campo de la base de datos de “private_key” a true para saber si ya tiene una clave privada el usuario. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Blockchain, API, Base de datos Firebase. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “gestión de contraseñas”. |
| | 3 | En un popup se pedirá al usuario la nueva clave privada maestra. |
| | 4 | El usuario rellena el campo de la clave y pulsa aceptar. |
| **Extensiones** | **Paso** | **Acción** |
| | 4 | El usuario deja en blanco el campo, por lo que no se puede aceptar la clave. |
| | 4 | El usuario vuelve a navegar a otra página antes de registrar la clave. |

*Tabla 7: Caso de uso: Creación de clave privada*

***Añadir contraseña***

| | | |
|---|---|---|
| **Caso de Uso CU05** | Añadir contraseña en la blockchain | |
| **Objetivo** | Se registra la nueva contraseña en un bloque de la blockchain, y se registra una nueva plataforma en la base de datos. | |
| **Precondiciones** | El usuario ya ha iniciado sesión, la aplicación web y base de datos están operativas y ya se ha registrado/insertado una clave privada maestra antes. | |
| **Postcondiciones** | Se crea un bloque de la blockchain con los datos de la nueva contraseña cifrados y se añade al campo de la base de datos “platforms” la nueva plataforma registrada. Se refresca la página para mostrar la contraseña añadida. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Blockchain, API, Base de datos Firebase. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “gestión de contraseñas”. |
| | 3 | El usuario ingresa o registra la clave privada maestra. |
| | 4 | El usuario pulsa el botón de añadir contraseña. |
| | 5 | El usuario rellena todos los campos que se piden; contraseña y plataforma, y pulsa aceptar. |
| **Extensiones** | **Paso** | **Acción** |
| | 5 | El usuario deja en blanco los campos, por lo que no se puede aceptar. |
| | 5 | El usuario cancela y vuelve hacia atrás. |

*Tabla 8: Caso de uso: Añadir contraseña*

***Modificar contraseña***

| | | |
|---|---|---|
| **Caso de Uso CU06** | Modificar contraseña en la blockchain. | |
| **Objetivo** | Se modifica una contraseña ya existente en la blockchain, y se cambia, si procede, la plataforma existente en la base de datos. | |
| **Precondiciones** | El usuario ya ha iniciado sesión, la aplicación web y base de datos están operativas, ya se ha registrado/insertado una clave privada maestra antes y se ha registrado al menos una contraseña antes. | |
| **Postcondiciones** | Se crea un bloque de la blockchain con los datos de la contraseña modificada encriptados y se cambia, si procede, al campo de la base de datos “platforms” en la plataforma modificada. Se refresca la página para mostrar los datos actualizados. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Blockchain, API, Base de datos Firebase. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “gestión de contraseñas”. |
| | 3 | El usuario ingresa o registra la clave privada maestra. |
| | 4 | El usuario pulsa el botón de modificar la contraseña. |
| | 5 | El usuario rellena todos los campos que quiere cambiar y pulsa “salvar”. |
| **Extensiones** | **Paso** | **Acción** |
| | 5 | El usuario deja en blanco los campos, por lo que no se puede aceptar. |
| | 5 | El usuario cancela y no hay más cambios. |
| | 5 | El usuario no realiza cambios y no se salvan los cambios. |

*Tabla 9: Caso de uso: Modificar contraseña*

***Eliminar contraseña***

| | | |
|---|---|---|
| **Caso de Uso CU07** | Eliminar contraseña. | |
| **Objetivo** | Se elimina la plataforma elegida de la base de datos y con ello se elimina. | |
| **Precondiciones** | El usuario ya ha iniciado sesión, la aplicación web y base de datos están operativas, ya se ha registrado/insertado una clave privada maestra y se ha registrado al menos una contraseña antes. | |
| **Postcondiciones** | Se quita la plataforma elegida de la base de datos y con ello se elimina. Se refresca la página para mostrar los datos actualizados. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Blockchain, API, Base de datos Firebase. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “gestión de contraseñas”. |
| | 3 | El usuario ingresa o registra la clave privada maestra. |
| | 4 | El usuario pulsa el botón de modificar la contraseña. |
| | 5 | El usuario pulsa el botón eliminar. |
| **Extensiones** | **Paso** | **Acción** |
| | 5 | El usuario cancela y no hay más cambios. |

*Tabla 10: Caso de uso: Eliminar contraseña*

***Consultar contraseñas cifradas***

| | | |
|---|---|---|
| **Caso de Uso CU08** | Consultar contraseñas cifradas. | |
| **Objetivo** | Mostrar al usuario sus contraseñas almacenadas en la blockchain | |
| **Precondiciones** | El usuario ya ha iniciado sesión, la aplicación web y base de datos están operativas, ya se ha registrado/insertado una clave privada maestra antes y se ha registrado al menos una contraseña antes. | |
| **Postcondiciones** | Se abre una ventana con la información en formato JSON de los bloques de la blockchain con la información del usuario. | |
| **Actores primarios** | Usuario. | |
| **Actores secundarios** | Aplicación web, Blockchain, API, Base de datos Firebase. | |
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “gestión de contraseñas”. |
| | 3 | El usuario ingresa o registra la clave privada maestra. |
| | 4 | El usuario pulsa el botón de ver contraseñas en la blockchain. |
| **Extensiones** | **Paso** | **Acción** |
| | 4 | El usuario no tiene contraseñas registradas y no se muestra el botón. |

*Tabla 11: Caso de uso: Consultar contraseñas cifradas*

***Generar contraseña aleatoria***

| | |
|---|---|
| **Caso de Uso CU09** | Generar contraseña aleatoria. |
| **Objetivo** | Mostrar al usuario una contraseña aleatoria segura para que la pueda copiar. |
| **Precondiciones** | El usuario ya ha iniciado sesión y la aplicación web está operativa. |
| **Postcondiciones** | Se genera una clave en formato texto para que el usuario pueda copiarla. |
| **Actores primarios** | Usuario. |

| **Actores secundarios** | Aplicación web. | |
|---|---|---|
| **Descripción** | **Paso** | **Acción** |
| | 1 | El usuario accede a la aplicación e inicia sesión. |
| | 2 | El usuario navega a la página de “generador”. |
| | 3 | El usuario pulsa en generar contraseña. |
| | 4 | El usuario puede copiar la contraseña mediante un botón. |

*Tabla 12: Caso de uso: Generar contraseña aleatoria*

## 3.7 Desarrollo

### 3.7.1 Frontend

El frontend se refiere a toda la parte de la aplicación con la que interactúa el usuario directamente, es decir, la interfaz de usuario. En el proyecto se ha usado la biblioteca de JavaScript React para desarrollar el frontend. Esta herramienta funciona por componentes, que son piezas de la interfaz gráfica reutilizables, haciendo que una pantalla de la aplicación web esté formada por varios componentes.

Para el desarrollo de la aplicación se utilizan los lenguajes de programación CSS, HTML y JavaScript para hacer el diseño, sin utilizar otros frameworks que existen para partir de diseños ya realizados por terceros.

***Idiomas***

Incluir idiomas en una aplicación web puede ser muy notable dependiendo de para qué público o país esté dirigida, en el caso de este proyecto, se ha realizado todos los textos en inglés con el objetivo de poder abarcar al mayor número de usuarios, ya que este idioma representa el más hablado del mundo.

Aunque por otro lado, todos los textos que aparecen en inglés, están traducidos al español, en un fichero llamado “localizable” en el mismo proyecto. Esto se ha desarrollado así para que en versiones futuras sólo haya que implementar una funcionalidad para cambiar de idioma y ahorrarse el proceso de traducir manualmente.

***Componentes principales***

La interfaz gráfica del proyecto está formada por estas pequeñas piezas de código, formando así toda la aplicación. Los principales componentes son:

- Navbar: es una barra localizada en todas las pantallas encargada de efectuar la navegación a las diferentes secciones de la aplicación


*Figura 18: Barra navegadora de la aplicación*

- locatorbar: es una barra lateral que se encarga de dotar a la aplicación de un diseño original, con la funcionalidad de localizar al usuario en qué página de la aplicación se encuentra, una forma más visual que el anterior componente.

*Figura 19: Barra lateral de localización de la aplicación*

- Botones: los botones son reutilizados en toda la aplicación para que sean similares entre sí y evitar la repetición de código.
- Footer: se localiza al final de la pantalla principal y su utilidad es informar al usuario de los derechos de autor y proporcionar la política de privacidad.

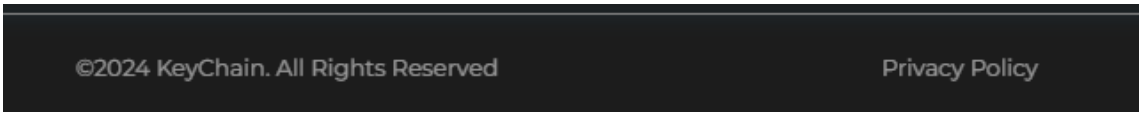


*Figura 20: Footer de la aplicación*

***Ventanas del proyecto***

La aplicación está formada por las siguientes ventanas principales.

| Nombre de la ventana | Descripción |
|---|---|
| **About** | Esta primera ventana muestra una primera presentación de la aplicación mediante un scroll, exponiendo información de las soluciones, servicios y fundadores del proyecto. No hace falta iniciar sesión para acceder a esta ventana. |
| **Login** | En esta pantalla se pide al usuario un correo y contraseña para iniciar sesión en la aplicación. O por el contrario, pulsar el botón de registrarse si no lo está aún. |
| **SignUp** | En esta pantalla se piden las credenciales para registrarse. O por el contrario, pulsar el botón de iniciar sesión si ya se tiene una cuenta. |

| | |
|---|---|
| **Home** | Esta pantalla es la principal cuando se inicia sesión en la aplicación, mostrando textos y un botón para ir a la ventana de gestionar las contraseñas. |
| **Manage** | Esta pantalla es la encargada de mostrar al usuario toda la interfaz de la gestión de contraseñas. Mostrando primeramente un campo para introducir la clave privada maestra, o por el contrario si no se ha creado, el registro de la misma. Después de realizar este paso, se muestran las contraseñas al usuario. |
| **PasswordGenerator** | En esta pantalla se muestra un botón y un campo donde aparecerá la contraseña segura generada aleatoriamente. |

*Tabla 13: Ventanas principales de la aplicación*

La ventana *Manage*, donde se gestionan las contraseñas, ofrece dos vistas: una en forma de bloque y otra en forma de lista, para facilitar la visualización de las claves del usuario.

En la vista de bloque, se presentan los siguientes datos: la plataforma, la contraseña oculta con asteriscos (que puede visualizarse haciendo clic en el botón de ojo), la fecha y hora en que se registró la contraseña y se minó el bloque (timestamp), y el proof o nonce (el número utilizado para minar el bloque).

Además, esta ventana cuenta con un botón que genera una vista flotante, la cual muestra al usuario cómo están almacenadas sus contraseñas en la blockchain.

Otra ventana importante es la de *Password generator*. Esta ventana, mediante un botón y la función 'Math.random()', genera una contraseña alfanumérica de 12 caracteres, asegurando así la aleatoriedad y seguridad de la contraseña creada.

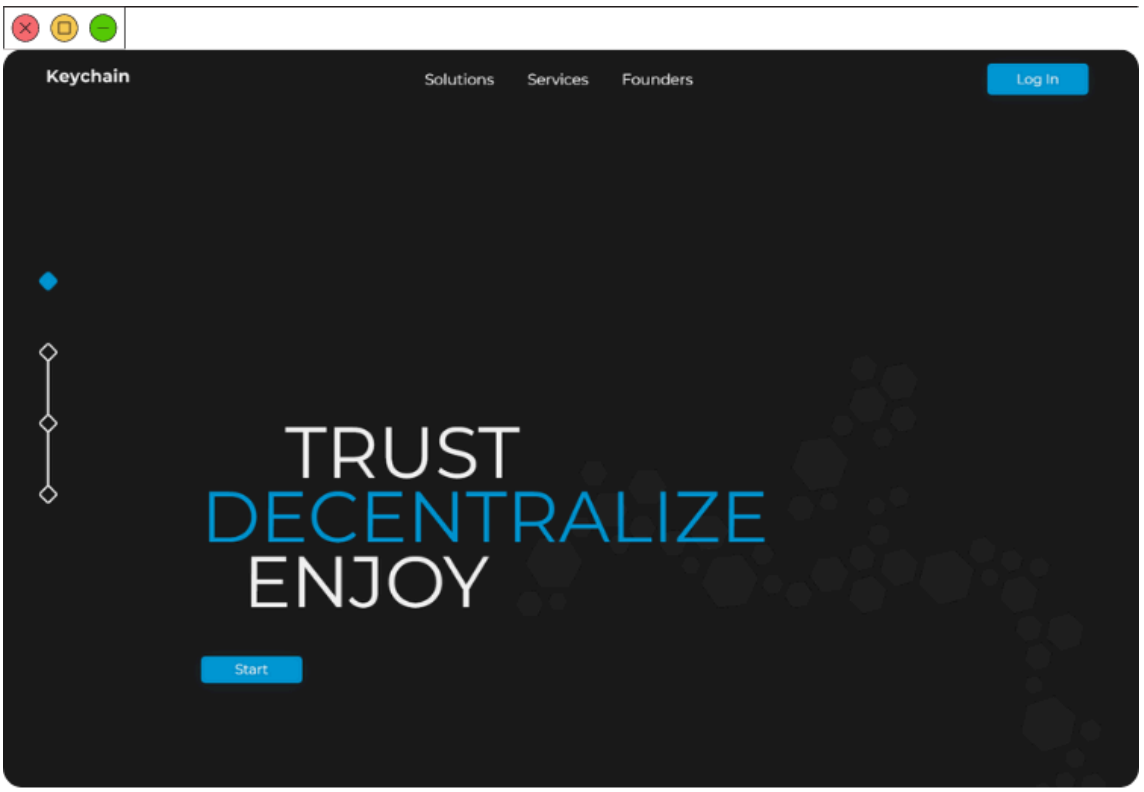


*Figura 21: About: ventana principal*

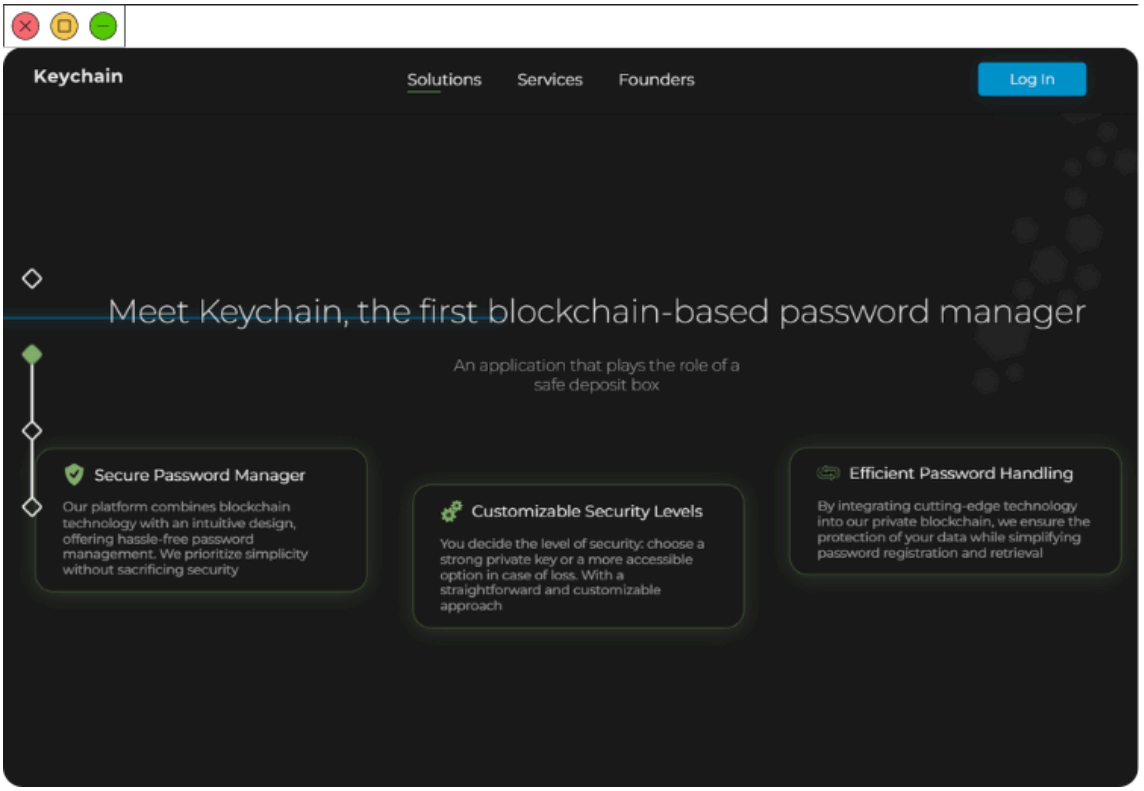


*Figura 22: About: ventana principal sección soluciones*

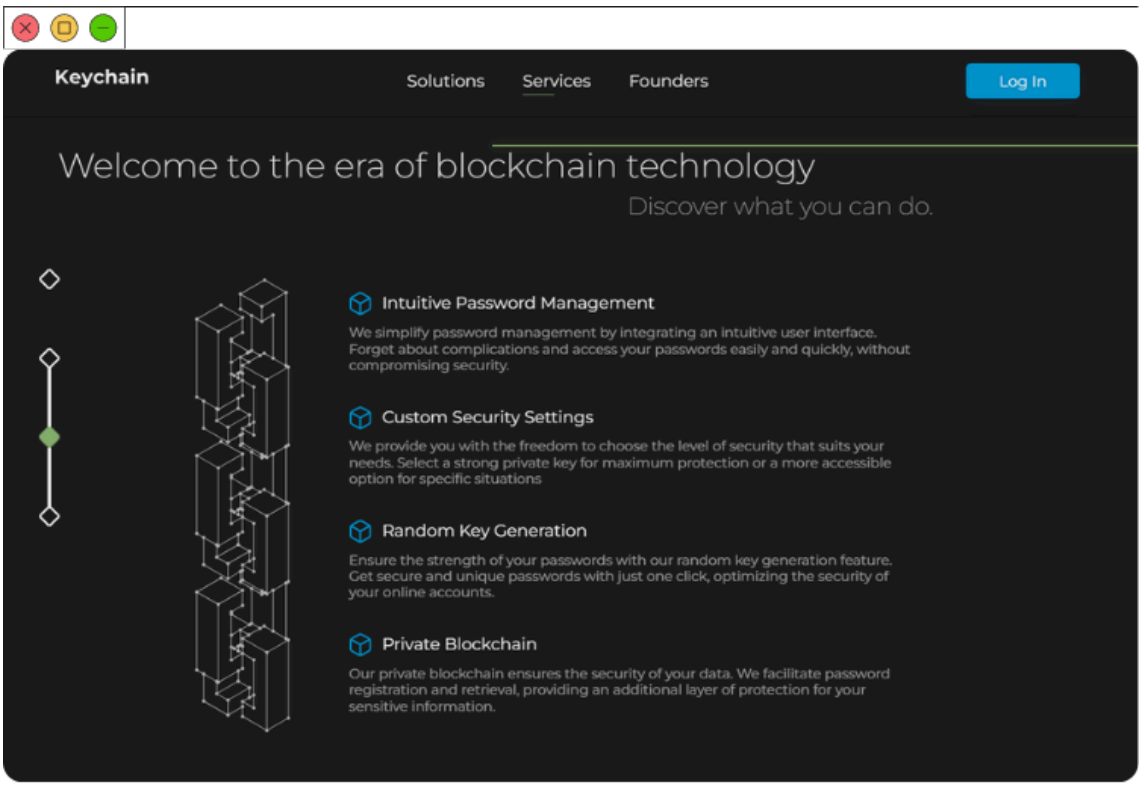


*Figura 23: About: ventana principal sección servicios*

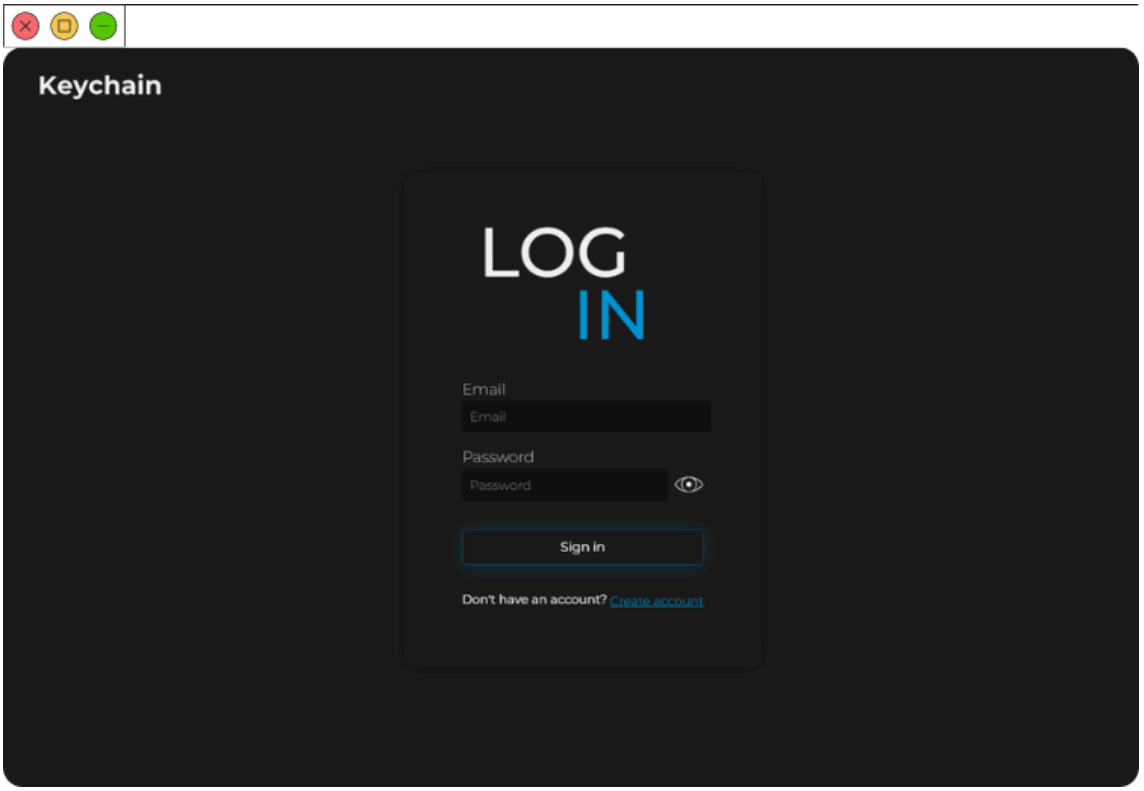


*Figura 24: Login: ventana de inicio de sesión*

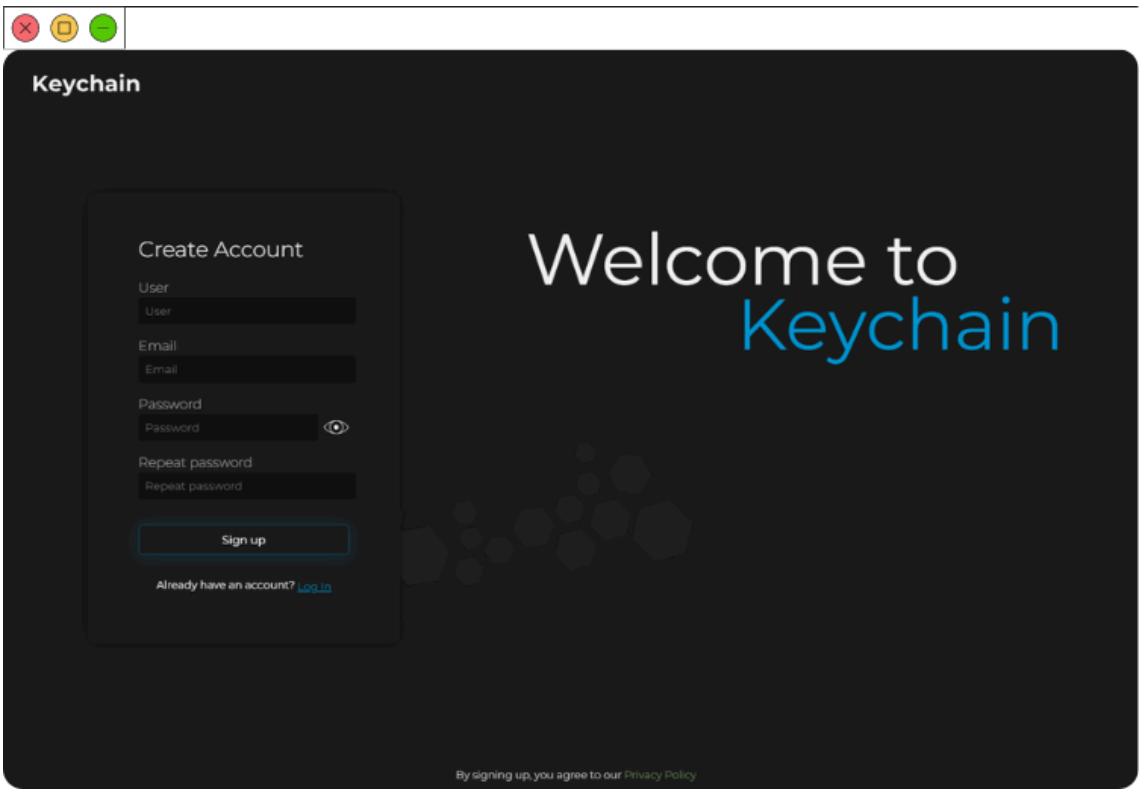


*Figura 25: SignUp: ventana de registro*

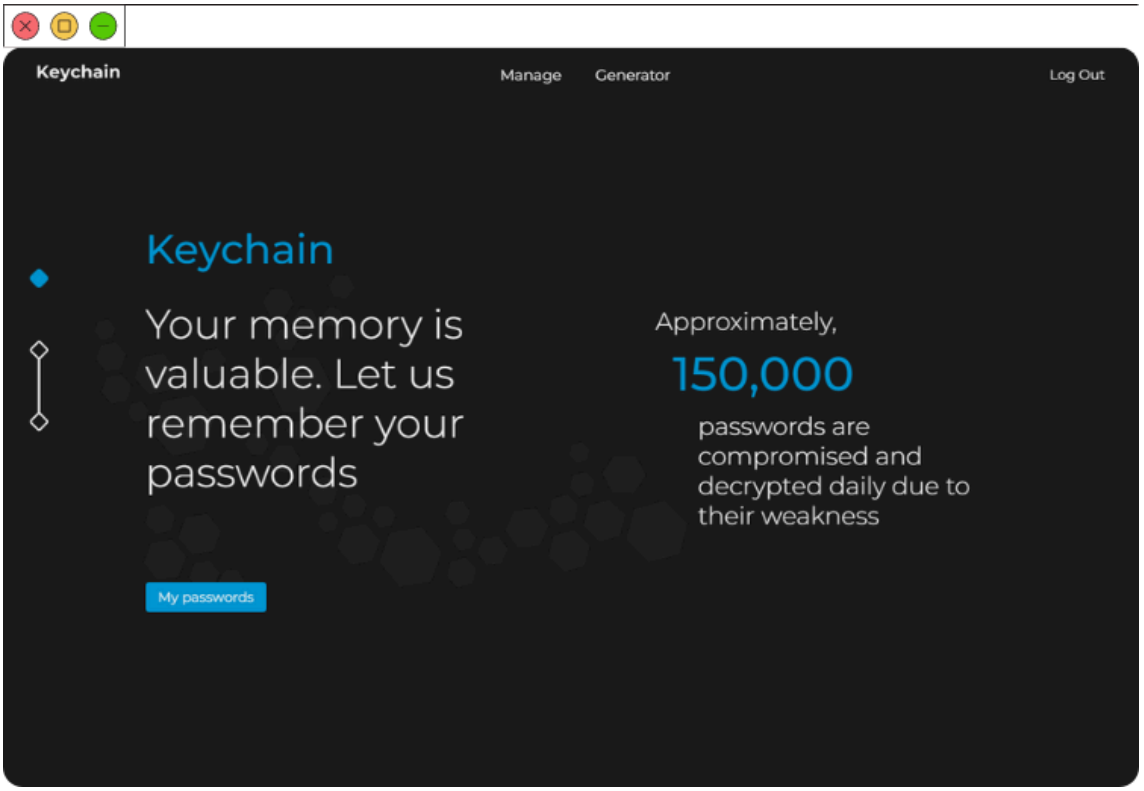


*Figura 26: Home: ventana de presentación*

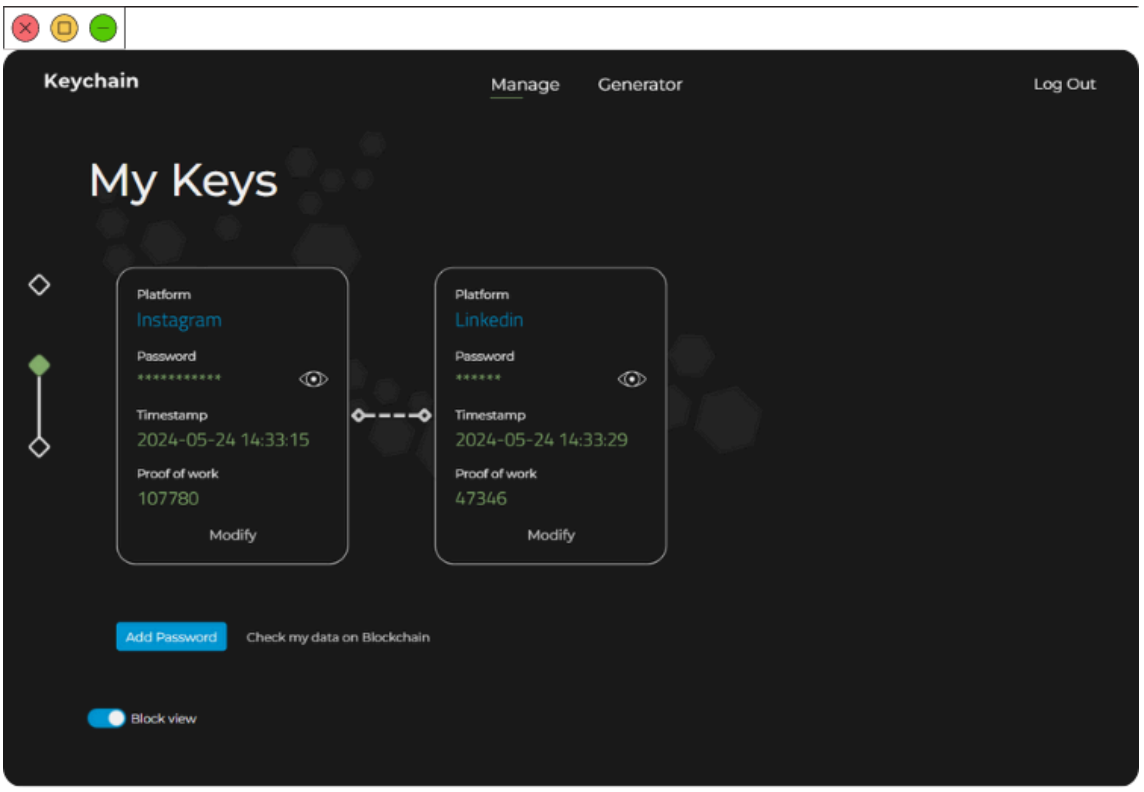


*Figura 27: Manage: ventana gestor de contraseñas*

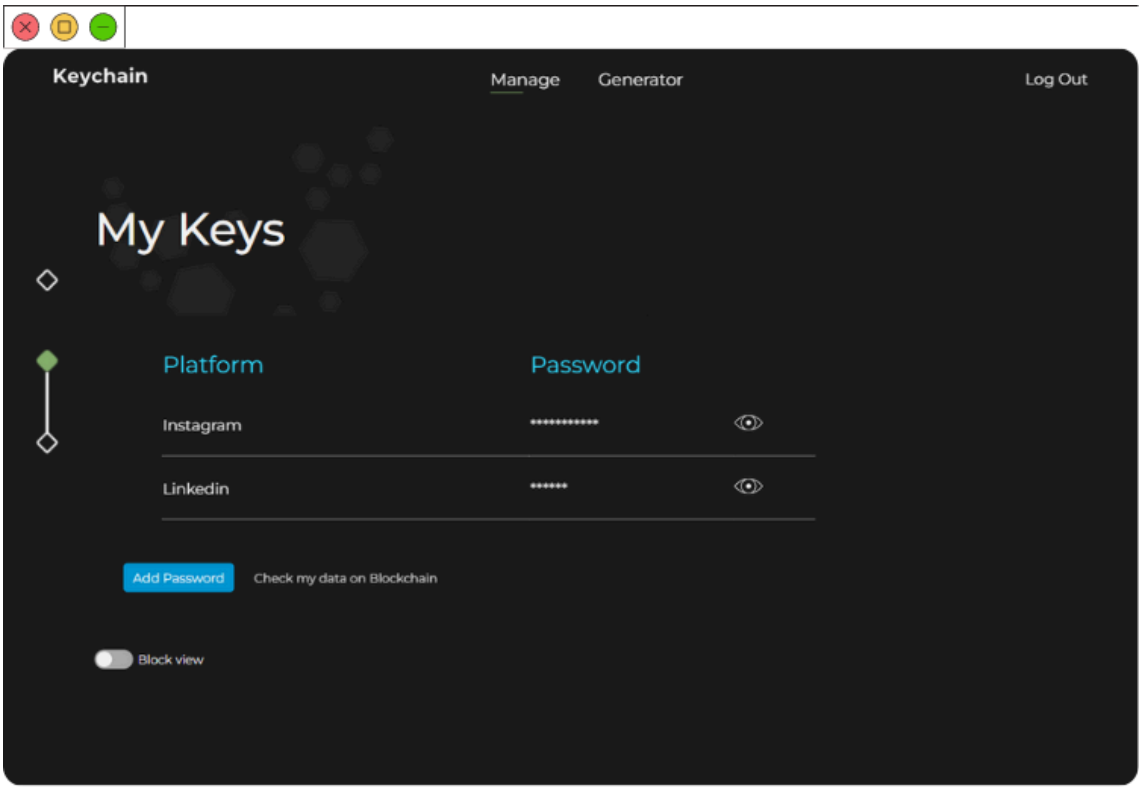


*Figura 28: Manage: ventana gestor de contraseñas (vista de lista)*

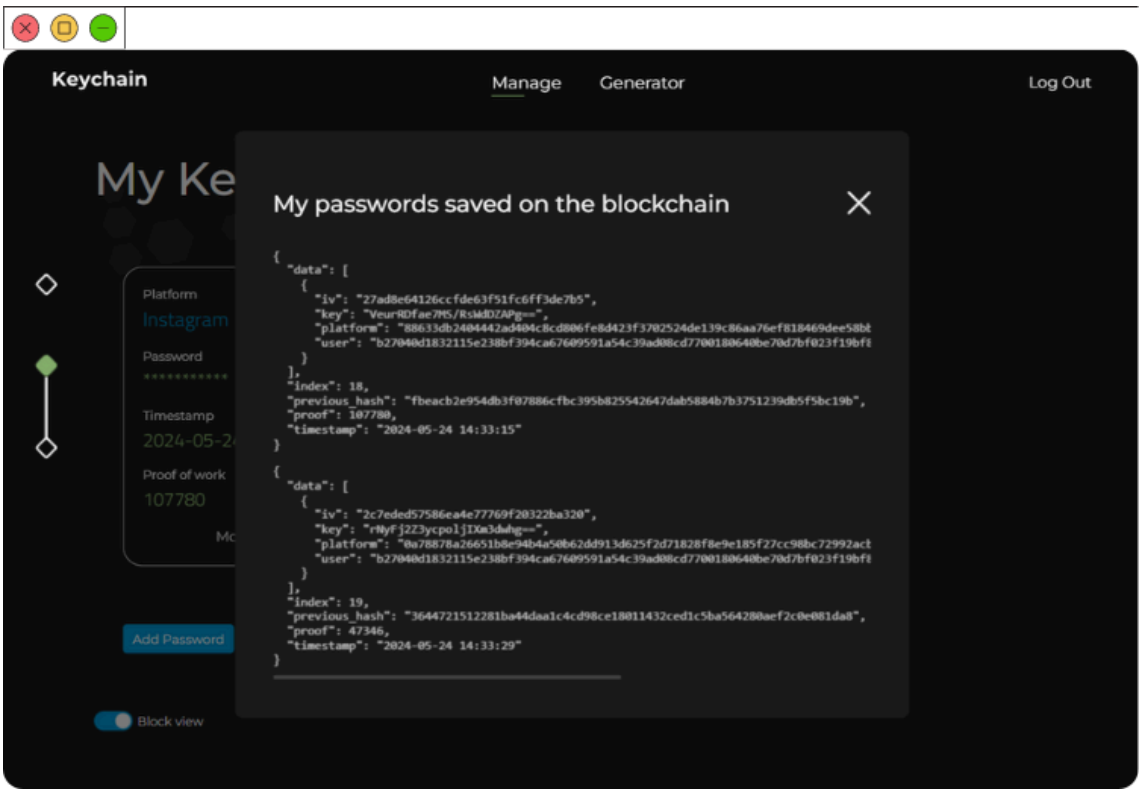


*Figura 29: Manage: ventana de contraseñas en la blockchain*

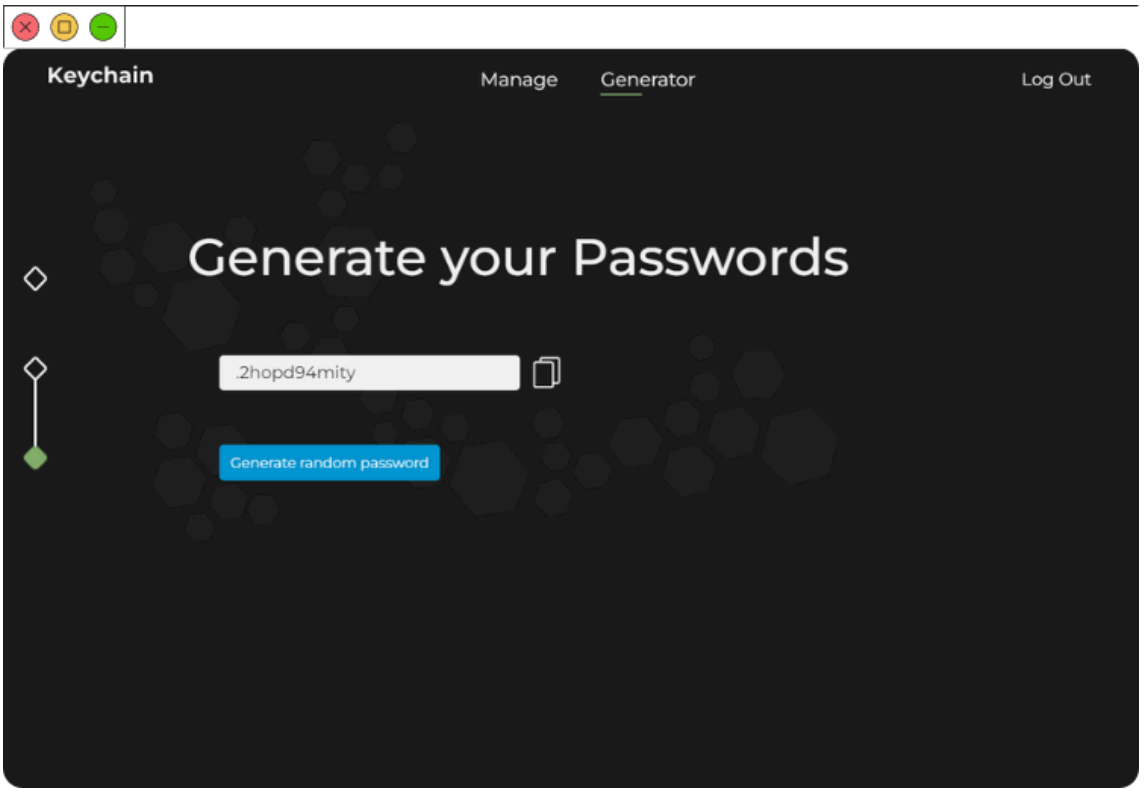


*Figura 30: PasswordGenerator: ventana de generador de contraseñas*

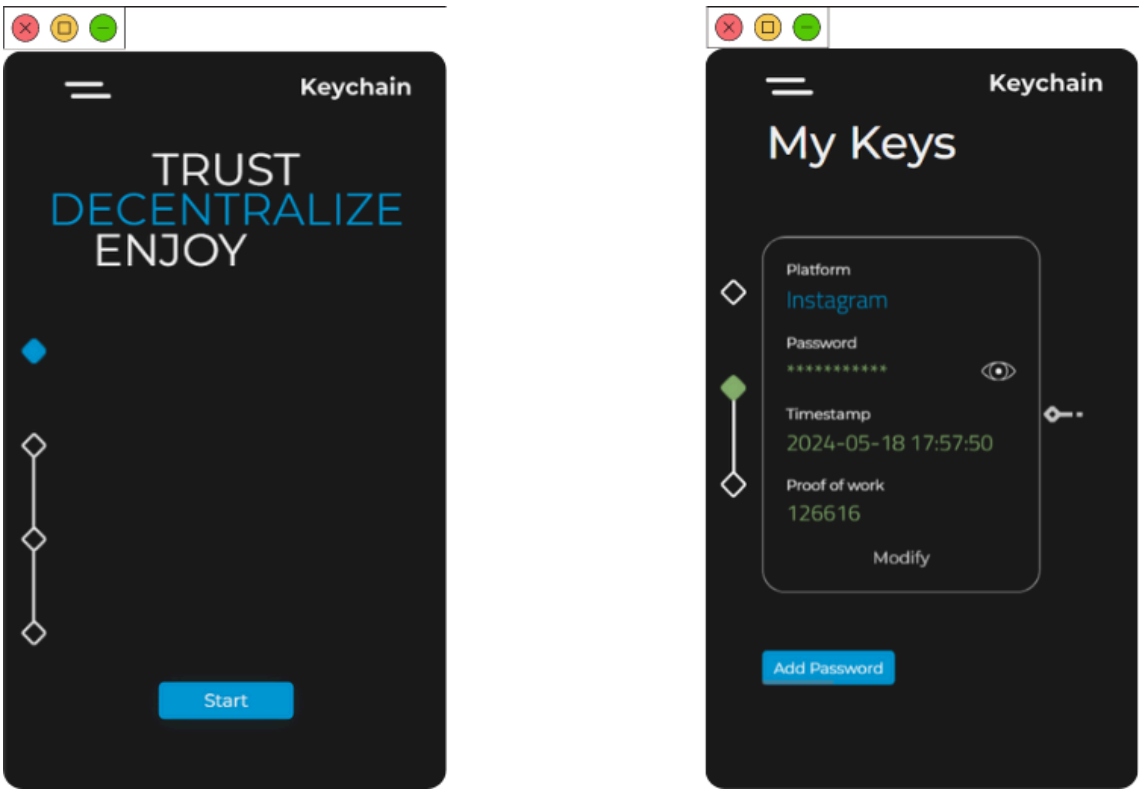


*Figura 31: Versión móvil: ventana principal y gestor de contraseñas*

### 3.7.2 Backend

El backend representa todos aquellos servicios que se localizan en un servidor separado de la interfaz gráfica, es decir, es el lugar donde se conecta el frontend para que este pueda tener funcionalidad. En este proyecto hace referencia a la API y la base de datos NoSQL.

***API***

La API está desarrollada en Python utilizando la herramienta Flask y se encarga de comunicarse con los nodos de la blockchain. Tiene varias funciones (endpoints) que ejecutan los métodos correspondientes de la blockchain. Las funciones de la API son las siguientes:

- *mine_block()*: Método GET que llama a la función de la blockchain para minar un bloque, devolviendo un mensaje de éxito y la información del bloque minado.
- *get_chain()*: Método GET que consulta toda la cadena de bloques y la devuelve en formato JSON.
- *is_valid()*, Método GET que verifica la validez de la cadena de bloques del nodo, devolviendo un mensaje que indica si es válida o no.
- *add_data()*, Método POST que llama a la función de la blockchain para registrar una nueva contraseña y posteriormente minar el bloque, devolviendo un mensaje de éxito o error.

- *add_user()*: Método POST que llama a la función de la blockchain para registrar un nuevo dato cifrado que verifica una clave privada maestra de un usuario, y luego mina el bloque. Devuelve un mensaje de éxito o error.
- *get_data()*: Método GET que, usando como parámetros la plataforma y el usuario, llama a la función de la blockchain para devolver el bloque con la información de la contraseña solicitada en formato JSON, si existe.
- *get_user():* Método GET que, usando como parámetro el UID del usuario, llama a la función de la blockchain para devolver el bloque con la información cifrada para verificar la clave privada maestra, en formato JSON, si existe.
- *connect_node():* Método GET que llama a la función de la blockchain para conectar entre sí los nodos de la red, devolviendo un mensaje de éxito o error.
- *replace_chain():* Método GET que llama a la función de la blockchain para asegurar que todos los nodos de la blockchain tengan la misma información, reemplazando la información de los nodos desactualizados o incorrectos. Devuelve un mensaje informativo sobre si se ha tenido que reemplazar la cadena o no.

***Base de datos***

La base de datos que se ha implementado en el proyecto es Firebase, una NoSQL documental desarrollada por Google y creada específicamente para aplicaciones web y móviles.

Esta herramienta se utiliza por un lado para la autenticación de los usuarios en la plataforma mediante una dirección de correo electrónico y una contraseña.

| Identificador | Proveedores | Fecha de creación ↓ | Fecha de acceso | UID de usuario |
|---|---|---|---|---|
| a@gmail.com | ✉ | 8 may 2024 | 8 may 2024 | LdpsT9bN1SOmy0oDfaafhDY... |
| guille@gmail.com | ✉ | 8 may 2024 | 8 may 2024 | mK0Na2FBj1Nc7tRJ42vyqXkI... |
| elenagb01@hotmail.com | ✉ | 4 may 2024 | 4 may 2024 | SEQw4JqC0oZVQ5K2ZO01Bq... |
| yago@gmail.com | ✉ | 17 abr 2024 | 20 may 2024 | 4YoiK7jZalR3JsGFQCfGrROW... |

*Figura 32: Ejemplo de los datos de autenticación en Firebase*

Y por otro lado, se utiliza para el almacenamiento de cuatro campos de datos del usuario; el UID, un nombre de usuario único, estado de su clave privada maestra, y las plataformas que el usuario almacena.

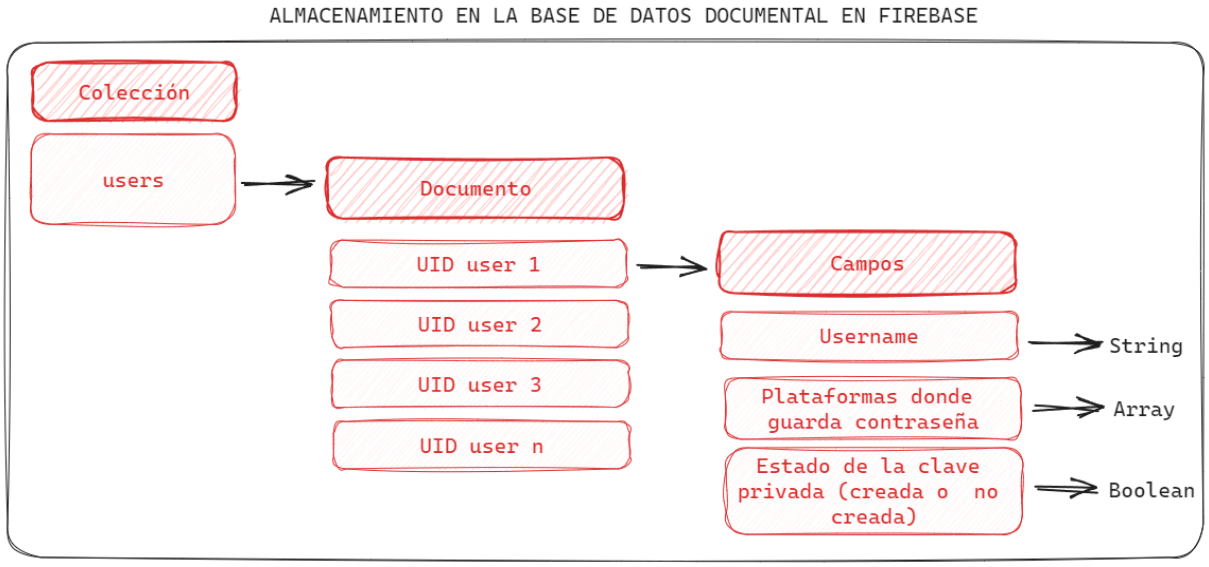


*Figura 33: Información en la base de datos NoSQL documental*

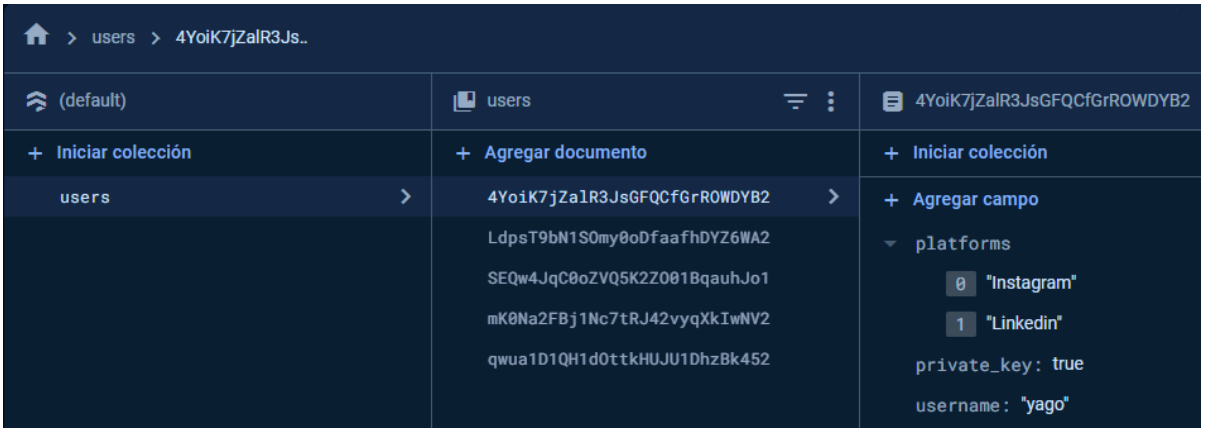


*Figura 34: Ejemplo de los datos de usuario en Firebase*

### 3.7.3 Blockchain

Para el proyecto se optó por el desarrollo de una blockchain privada, ya que con ello se conseguía una personalización y seguridad mayor. Está realizada en el lenguaje de programación Python y cuenta con ciertos métodos desarrollados específicamente para el funcionamiento de la aplicación.

***Datos de la blockchain***

Existen dos tipos de bloques que almacenan datos del usuario: los que se utilizan para almacenar las contraseñas con sus datos correspondientes y los que verifican que se ha introducido la clave privada maestra correcta.

- El primer tipo, que guarda las contraseñas, tiene los siguientes campos: la plataforma hasheada, el UID del usuario hasheado, la contraseña cifrada, y el IV.
- El segundo tipo, que se utiliza para comprobar la clave privada maestra del usuario, tiene los siguientes campos: UID del usuario, la información cifrada para comprobar la validez de la clave, el "IV" y el "salt".

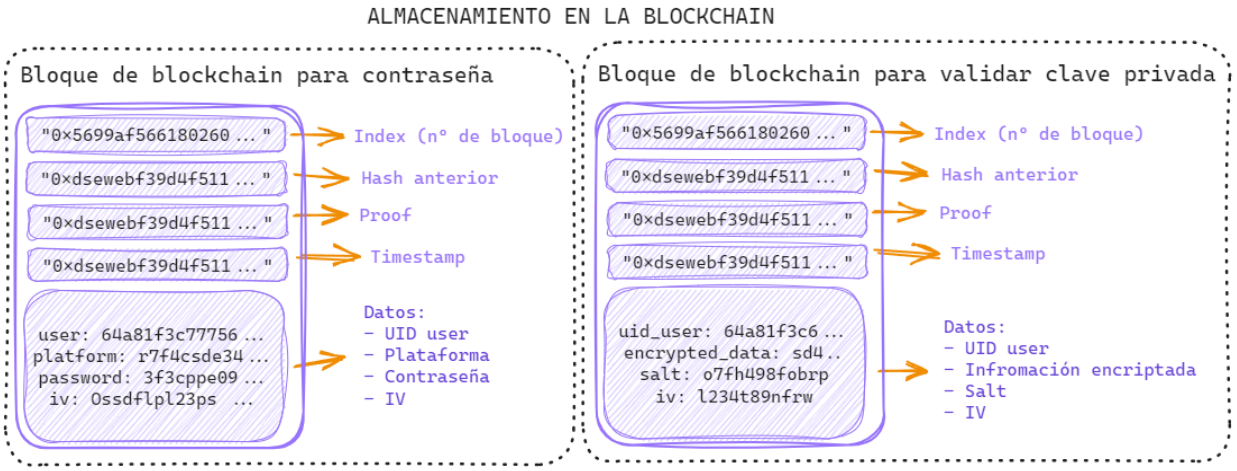


*Figura 35: Arquitectura de los bloques de la blockchain*

Por otro lado, existen los datos genéricos de cada bloque, que le hacen único, estando formado por los siguientes campos: una marca de tiempo (*'timestamp'*), un número con el que se ha conseguido minar el bloque (*'proof'* o *'nonce'*), el hash del anterior bloque (*'previous_hash'*) y el número del bloque (*'index'*). En el caso de la siguiente figura, se aprecia un bloque que guarda la contraseña de una plataforma de un usuario.

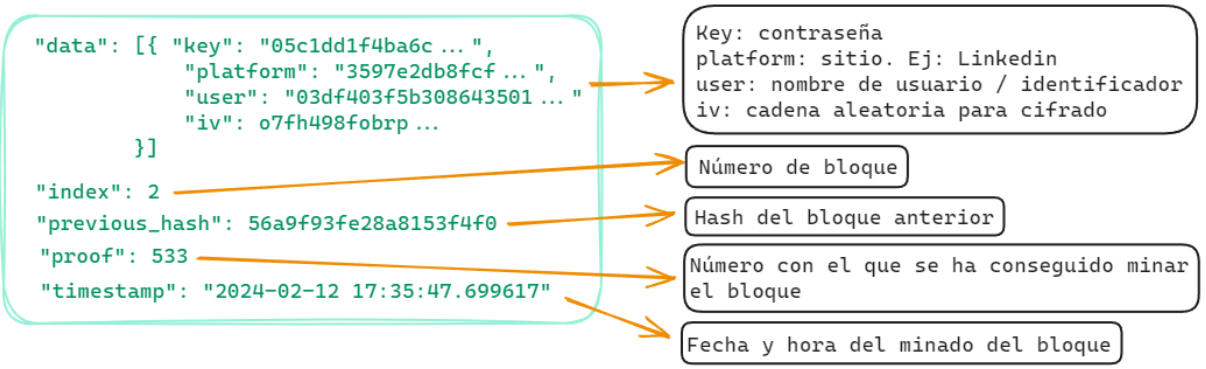


*Figura 36: Información del bloque almacenando una contraseña*

***Composición***

A continuación se muestra cómo está compuesta la cadena de bloques, realizando en primer lugar una inicialización de la blockchain cuando esta se ejecuta por primera vez.

Inicialización:

- **Datos**: Se crean dos listas vacías: una para la cadena de bloques (‘*chain’*) y otra para los datos (‘*data’*).
- **Carga de Cadena**: Se verifica si existe el archivo ‘*chain.json’*. Si existe, se carga la cadena de bloques desde este archivo; si no, se crea el primer bloque.
- **Nodos**: Se inicializa la variable ‘*nodes’* para almacenar los nodos existentes en la red.

Métodos:

*create_block(self, proof, previous_hash):*

- Crea un bloque con la información pendiente de la lista *‘data’*.
- Inserta el nuevo bloque en la lista *‘chain’*.

*get_previous_block(self):*

- Retorna el bloque anterior al actual.

*proof_of_work(self, previous_proof):*

- Realiza la prueba de trabajo para minar un bloque.
- Calcula el *‘proof’* o *‘nonce’* recorriendo un bucle hasta que el hash SHA256 del *‘proof’* actual junto con el *‘previous_proof’,* tenga cuatro ceros al principio.
- Incrementa el *‘proof’* hasta encontrar el adecuado.

*hash(self, block):*

- Retorna el hash del bloque con la función SHA256.

*is_chain_valid(self, chain):*

- Valida la cadena de bloques.

- Verifica que no haya datos corruptos comprobando el *'previous_hash'* y *'proof'* de cada bloque.

*add_data(self, user, platform, key, salt):*

- Inserta los datos recibidos como parámetros en la lista *'data'*.
- Los datos recibidos son la información de una contraseña.

*add_user(self, uid_user, encrypted_data, salt, iv):*

- Inserta los datos de verificación de la clave privada recibidos como parámetros en la lista *'data'*.

*add_node(self, address):*

- Añade los nodos recibidos a la variable *'nodes'*.

*get_data(self, user, platform):*

- Consulta la blockchain para encontrar los datos que coinciden con el usuario y la plataforma recibidos como parámetros.
- La información devuelta es de contraseñas.
- La búsqueda se realiza en orden de actualidad - antiguo

*get_user(self, uid_user):*

- Consulta la blockchain para encontrar los datos que coinciden con el UID del usuario recibido como parámetro.
- La información devuelta es de verificación de la clave privada.
- La búsqueda se realiza en orden de actualidad - antiguo

*replace_chain(self):*

- Recorre la lista de nodos 'nodes' para identificar la cadena más larga.
- Reemplaza las blockchains desactualizadas con la cadena más larga.

*save_chain(self, file_name):*

- Guarda la cadena de bloques *'chain'* en un archivo JSON.

*upload_chain(self, file_name):*

- Reemplaza el archivo *'chain.json'* con la información actualizada de la cadena de bloques.

*(El código de los métodos mencionados se pueden consultar en el anexo)*

### 3.7.4 Algoritmos aplicados

El empleo de algoritmos en este proyecto es crucial para asegurar la seguridad de los datos del usuario. Sin un tratamiento adecuado, cualquier persona con acceso a la blockchain podría visualizar las contraseñas en texto claro.

***Cifrado/Descifrado de contraseñas***

En primer lugar, las contraseñas registradas por el usuario deben ser cifradas utilizando una clave privada maestra, la cual solo el usuario debe conocer. Esta clave tiene la función de cifrar y descifrar las contraseñas, pero no se guarda en ningún lugar. Si el usuario la pierde, perderá la capacidad de consultar sus contraseñas.

La clave privada maestra se somete a una función de derivación de clave (KDF), como PBKDF2, que transforma la clave original en una cadena de caracteres más larga y segura. Adicionalmente, a esta función se añade un “salt”, que es una cadena de caracteres aleatorios para mejorar la aleatoriedad del resultado de la función.

El siguiente paso es combinar el texto a cifrar con un vector de inicialización (IV) formado por caracteres aleatorios. Esto añade aleatoriedad al resultado y oculta patrones repetitivos en los datos originales.

Una vez se tiene el resultado de la función KDF y el mensaje que se quiere ocultar, se cifran los datos utilizando el algoritmo de cifrado simétrico AES, conocido por su seguridad. Se aplica el modo CBC por defecto, que oculta patrones repetitivos y mejora la seguridad general del cifrado.

Con este proceso, se asegura la seguridad de las contraseñas almacenadas en la blockchain, ya que se encuentran cifradas y protegidas mediante algoritmos robustos.

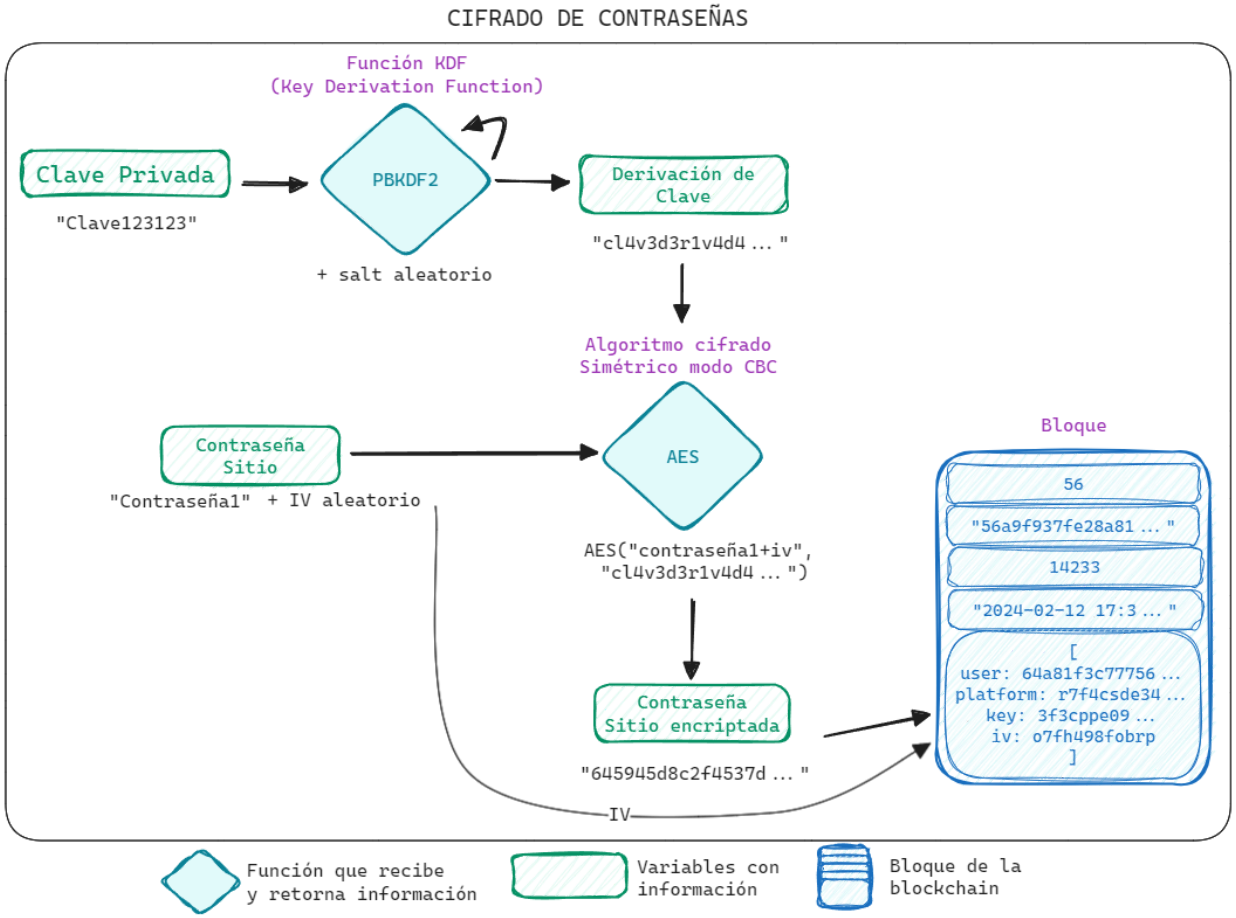


*Figura 37: Flujo de cifrado de contraseñas*

El descifrado es muy parecido al cifrado, pero a la inversa, se consulta la contraseña en la blockchain, y con ayuda de nuevo de la clave privada maestra y aplicando el KDF, ya tenemos la "llave" que descifra la información consultada.

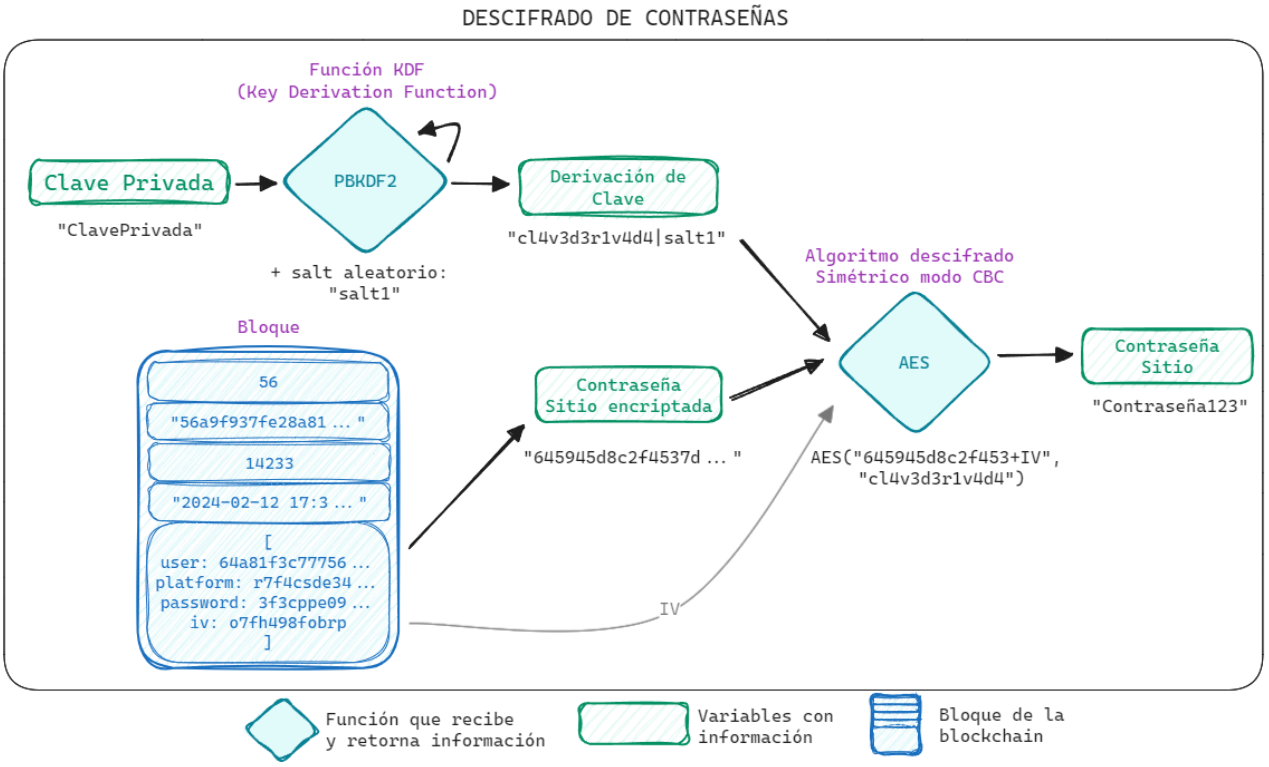


*Figura 38: Flujo de descifrado de contraseñas*

***Registro y validación de la clave privada maestra***

Para garantizar la seguridad en la consulta o registro de contraseñas, el usuario debe ingresar primero su clave privada maestra y autenticarse con ella. Esto permite cifrar y descifrar sus datos posteriormente. La clave privada maestra no se guarda en ningún lugar, por lo que su registro requiere un proceso específico.

Para registrarla, se cifra un dato particular (el hash del nombre de usuario) con la clave privada maestra, y el valor resultante de la operación se almacena en un bloque de la blockchain con los otros campos necesarios. Más tarde, se valida realizando el mismo procedimiento.

Para este proceso, se utilizan los bloques de autenticación, que recordando el apartado de blockchain, existen los campos: uid_user (el hash del UID del usuario registrado), encrypted_data (la información cifrada para la validación), el IV (para dotar a la función AES de más seguridad y aleatoriedad) y el salt (cadena de caracteres aleatoria, que al igual que el IV se utiliza para dar seguridad a la función de derivación de clave PBKDF2).

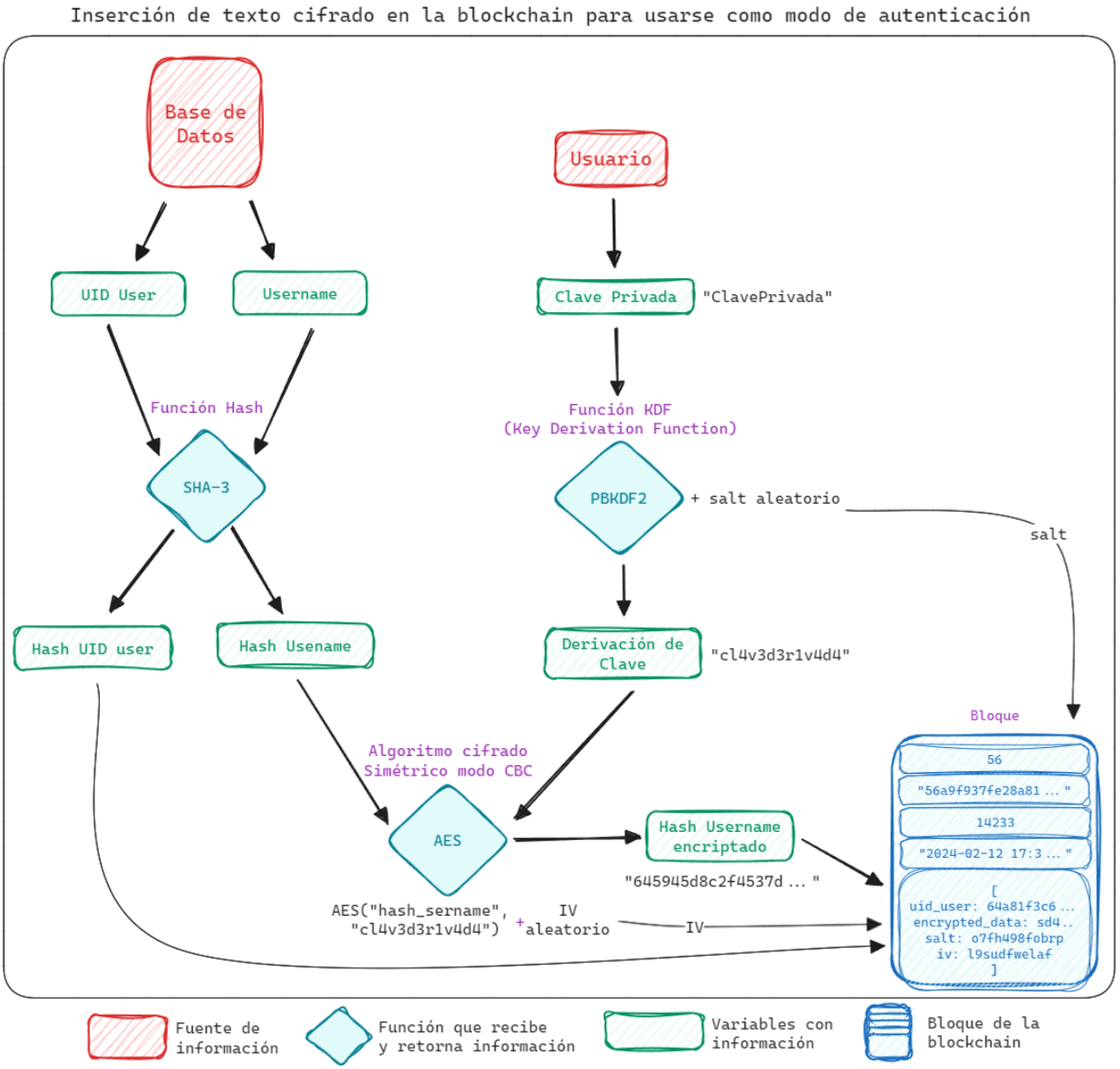


*Figura 39: Flujo de registro de información cifrada en un bloque*

Para la validación de que una clave privada es legítima, se realiza el procedimiento de descifrar la información almacenada en la blockchain. Si los datos cifrados coinciden, significa que la clave privada se ha insertado correctamente. De lo contrario, si no coinciden, se ha introducido una clave errónea que no puede descifrar los datos adecuadamente. Este proceso garantiza que solo el usuario autorizado pueda acceder a sus contraseñas.

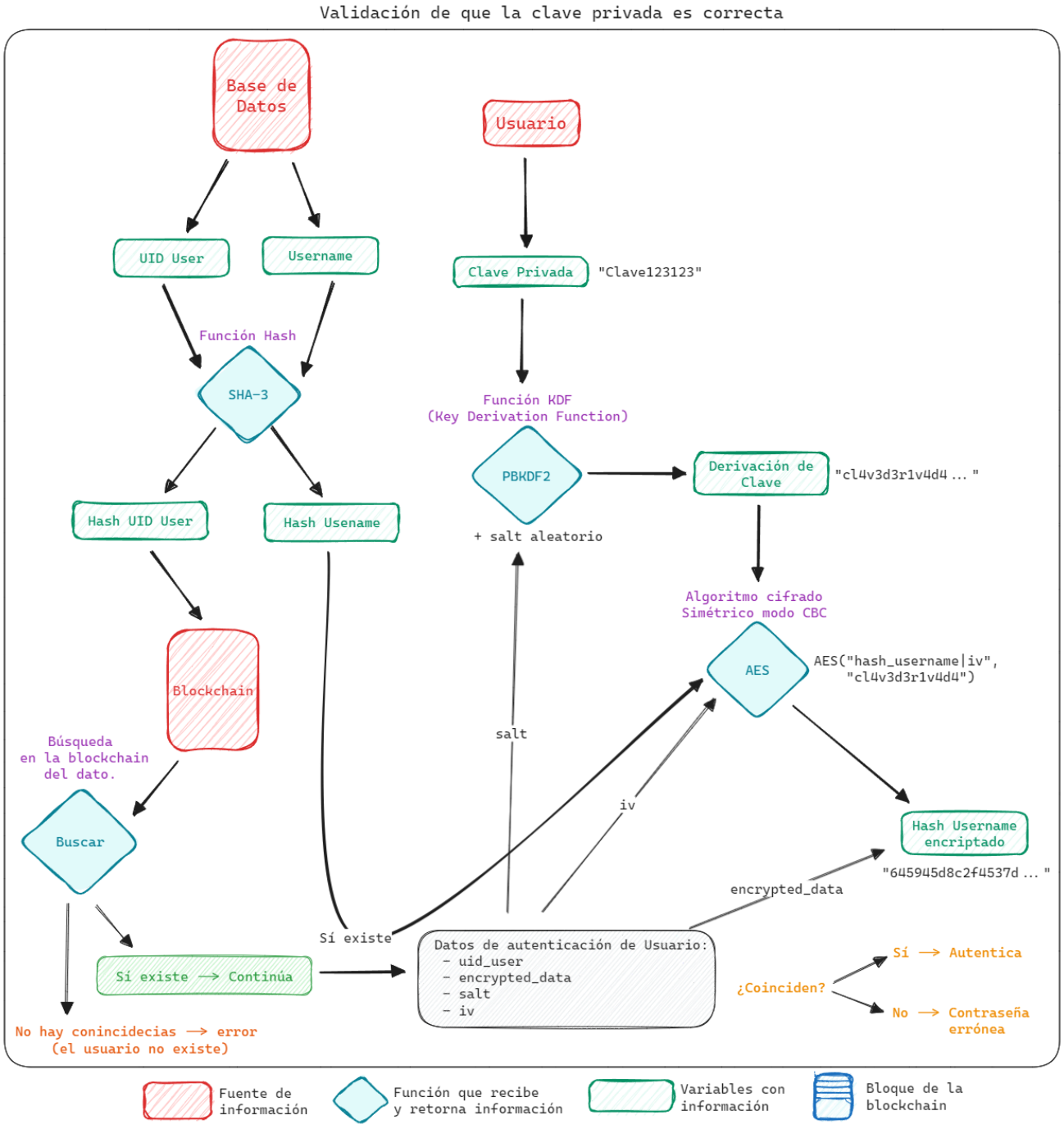


*Figura 40: Flujo de validación de la clave privada maestra del usuario*

***Registro y consulta de las contraseñas del usuario***

El almacenamiento y gestión de contraseñas en la blockchain requiere un procedimiento especial para el registro y la consulta, dado que todas las contraseñas están almacenadas en la blockchain, es necesario identificar cuál pertenece a cada usuario. Para lograr esto, cada bloque de la blockchain guarda la información necesaria para identificarlas de manera precisa.

Cuando las contraseñas son transmitidas a través de la red, se aplica un tratamiento para que el texto sea ilegible, pero pueda ser recuperado por el usuario legítimo una vez consultadas. Este proceso garantiza la confidencialidad de las contraseñas mientras están en tránsito.

La aplicación implementa las medidas de seguridad adecuadas, especialmente en lo referente al tráfico de información entre el frontend y el backend durante el registro y la consulta de contraseñas. Dado que la información puede ser interceptada por terceros mientras viaja por la red, es fundamental que los datos estén cifrados o sean ilegibles para garantizar su privacidad y seguridad.

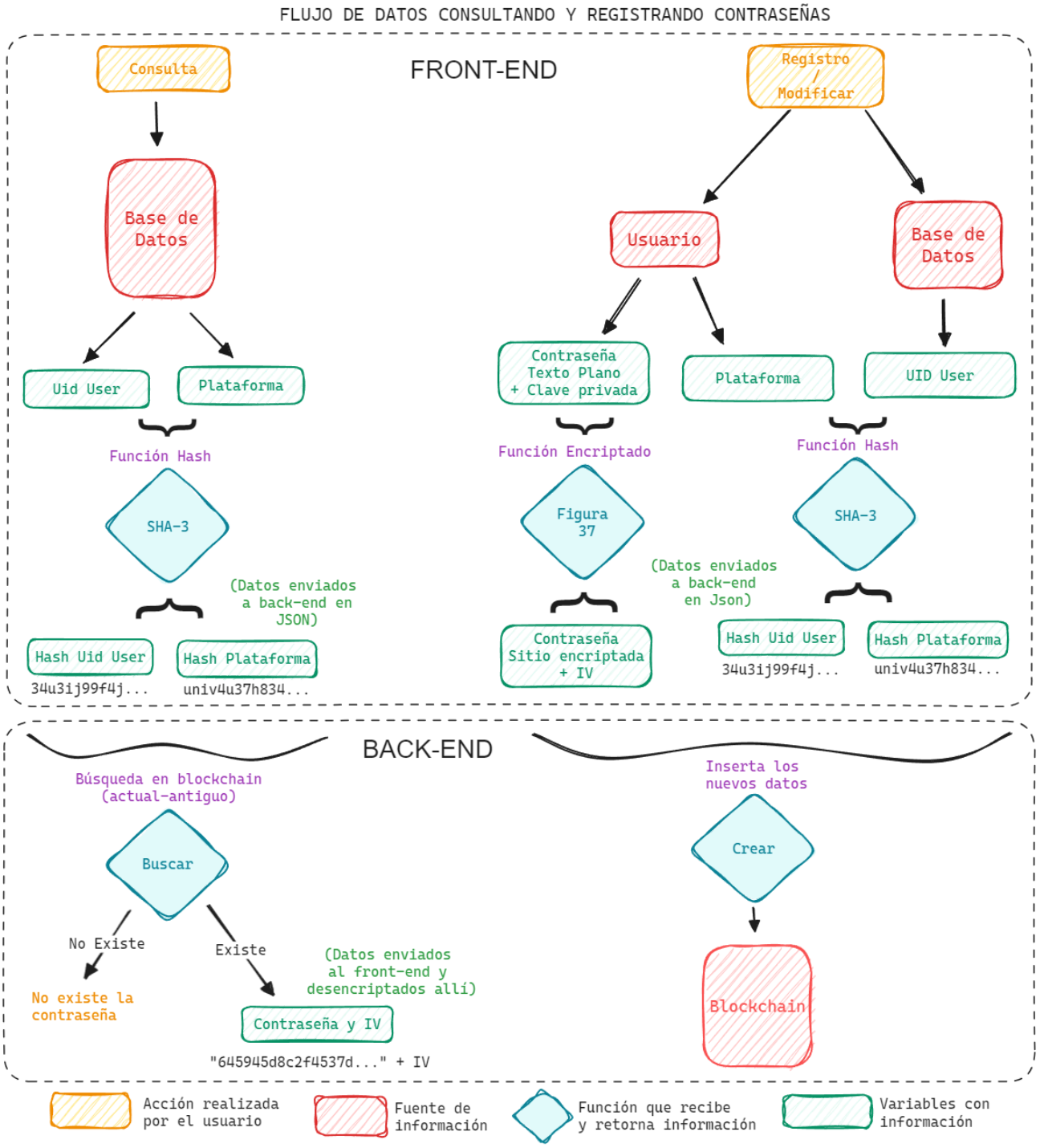


*Figura 41: Flujo de datos consultando y registrando contraseñas*

## 3.8 Plan de pruebas

### 3.8.1 Objetivo y alcance del plan de pruebas

El objetivo del plan de pruebas es garantizar la calidad y funcionalidad del software. Incluye la identificación de los tipos de pruebas a realizar, como pruebas funcionales, de usabilidad, de rendimiento y de seguridad, así como los entornos de prueba. Su propósito es asegurar que todos los componentes de la aplicación funcionan correctamente y cumplen con los requisitos especificados.

El plan de pruebas elaborado abarca la validación de funcionalidades clave como la generación, almacenamiento y recuperación de contraseñas, asegurando su cifrado y seguridad en la cadena de bloques. Incluirá pruebas de usabilidad e integridad de datos.

### 3.8.2 Estrategia de pruebas

Cada tipo de prueba tiene un cometido y para ello distinguimos en este plan los siguientes:

- **Pruebas Unitarias**: Pruebas de componentes individuales del sistema, como funciones y métodos en el backend (Python/Flask).
- **Pruebas de Integración**: Pruebas que verifican que los diferentes módulos del sistema interactúan correctamente, como la integración entre el frontend y el backend, entre el backend y la blockchain.
- **Pruebas de Rendimiento**: Pruebas para comprobar que los tiempos de respuesta están dentro de los límites aceptables.
- **Pruebas de Seguridad**: Pruebas que aseguran que el sistema es seguro y protege adecuadamente los datos sensibles, especialmente las contraseñas.

### 3.8.3 Detalle de las pruebas

Las pruebas que se llevan a cabo incluyen la verificación del correcto funcionamiento de la API, blockchain, la integración entre el frontend y el backend, y por último, la comprobación del cifrado de las contraseñas almacenadas.

***Pruebas de la API***

Para realizar las pruebas sobre la API, se utiliza una aplicación software (Postman) que permite hacer peticiones a los endpoints de la API, mostrando los resultados rápidamente.

Estas verificaciones consisten en hacer peticiones a los distintos métodos y comprobar uno a uno si la información devuelta por esta es correcta, y si se ha realizado en un tiempo aceptable para un usuario (entre 100 y 300 ms).

***Pruebas de la blockchain***

Las pruebas llevadas a cabo sobre la blockchain consisten en comprobar que las anteriores pruebas efectuadas en la API, entre las cuales se encuentran insertar y modificar datos en la blockchain, han funcionado correctamente.

Para ello, el archivo '*chain.json*', debe contener los datos insertados, por lo que se verifica que esta información se ha almacenado adecuadamente.

***Pruebas de integración***

En cuanto a las pruebas de integración entre los distintos módulos del proyecto, se ha realizado las verificaciones sobre la propia aplicación web, con ello se asegura que las distintas partes funcionan adecuadamente en conjunto.

Estas pruebas consisten en verificar todas las funcionalidades que conlleven la comunicación entre frontend y backend, es decir, la aplicación web con la que el usuario interactúa, la API, la blockchain y la base de datos Firebase. Las funcionalidades incluyen registrar al usuario, iniciar sesión, consultar, añadir, modificar y eliminar contraseñas.

***Pruebas de seguridad***

Las últimas pruebas elaboradas, se realizan en relación a la seguridad del cifrado de las contraseñas, comprobando un bloque de la blockchain que contenía datos de una contraseña.

El bloque debía pasar una serie de verificaciones hasta confirmar que el cifrado es lo suficientemente robusto y seguro, para ello, se examinan los campos del bloque para revisar que no son legibles, además, de que el dato cifrado sea de 128 bits (es como cifra el algoritmo AES-256) y que el IV es de 128 bits (para un cifrado aleatorio y seguro).

Por último, se intenta descifrar la contraseña con la clave privada maestra para comprobar el correcto funcionamiento del cifrado y descifrado.

*(Los casos de prueba se pueden consultar en el anexo)*

## 3.9 Resultados del proyecto

En este apartado se expone un breve resumen de los resultados obtenidos tras el desarrollo del proyecto, mencionando la evolución de los objetivos durante el transcurso del proyecto así como las conclusiones de las pruebas realizadas en la aplicación.

### 3.9.1 Objetivos alcanzados

En la aplicación, se ha logrado correctamente alcanzar el objetivo general que se ha propuesto, teniendo como resultado una plataforma funcional y poniendo el foco en la transparencia y la seguridad, con las funcionalidades correspondientes indicadas en el documento presente.

***Variaciones en los objetivos***

Los objetivos propuestos al inicio del proyecto tenían una razón justificada por los que se eligieron, aunque según el proyecto está en desarrollo, estas ideas pueden variar en mayor o menor medida.

En el caso de este proyecto, los objetivos no han tenido un cambio sustancial, pero sí se han tenido que alterar algunos aspectos para asemejarse más al propósito de la aplicación, que es la seguridad y privacidad, haciéndolo superior al principalmente diseñado. Los escasos cambios realizados tienen una justificación firme detrás.

En primer lugar, hay una funcionalidad en los objetivos iniciales que quizás llama la atención que no se hayan mencionado es este documento: la implementación de un “seguro” para la recuperación de la clave privada maestra del usuario. Aunque esta función implique una mayor comodidad del usuario, la razón por la que esta no se ha nombrado es la seguridad, ya que cuando un cliente quiere recuperar una clave privada la cual es la “llave” para descifrar todas sus contraseñas, esta se tiene que almacenar en algún sitio de la red, siendo por ello un punto de inseguridad añadido.

Además, no guardar esta clave por parte de la aplicación trae consigo un mayor grado de confianza por el usuario, ya que el único que tiene la posibilidad de ver sus contraseñas es el propio usuario.

Por otro lado, también se ha añadido una funcionalidad que no estaba en el plan inicial del anteproyecto, siendo esta la opción de consultar los bloques de la blockchain, que almacenan las contraseñas del usuario. Esta función, se decidió implementar debido a la imagen que quiere representar la aplicación en cuanto a la privacidad. Así, el usuario puede comprobar cómo y dónde se están almacenando sus contraseñas en todo momento.

Otro aspecto que no se menciona en el anteproyecto y sí en esta última versión, es la capacidad que tiene la aplicación web, que puede adaptarse a muchos tipos y tamaños de pantallas, teniendo así la cualidad de ser “responsive”.

### 3.9.2 Pruebas y validaciones

Las pruebas llevadas a cabo en el apartado de "plan de pruebas" han sido generalmente satisfactorias, con una base firme de validaciones realizadas en la aplicación. Durante el periodo de pruebas, se ha comprobado el correcto funcionamiento de los componentes más importantes que forman el proyecto, y más tarde el conjunto de los componentes funcionando como uno solo.

***Pruebas Unitarias***

Se validaron las funciones de la API para comunicarse con la blockchain correctamente. No se obtuvo ningún resultado inesperado, debido a que este módulo se puso a prueba severamente durante el desarrollo de la plataforma.

***Pruebas de Integración***

La interacción entre el frontend, backend y la blockchain fue evaluada y demostraron que los módulos se integran de manera fluida, sin problemas generales de comunicación o sincronización.

Sin embargo, una prueba en concreto de esta sección mostró un error que no se tuvo en cuenta durante el desarrollo del proyecto: no se verificaba que cuando un usuario modificaba su contraseña, esta no podía tener el mismo nombre de plataforma que otra que ya tuviese registrada, por lo que se corrigió correctamente.

***Pruebas de Rendimiento***

El sistema mostró un rendimiento estable, con tiempos de respuesta dentro de los límites aceptables. Las pruebas midieron mediante el software Postman la media del tiempo de respuesta de la API, dando como respuesta después de dieciocho intentos (probando tres veces cada método) una media de 96 ms, estando dentro de los límites aceptables.

***Pruebas de Seguridad***

En estas pruebas se confirma que el cifrado de contraseñas y la protección de datos son robustos. Se verificaron correctamente los siguientes puntos:

- Las contraseñas se cifran y descifran correctamente.
- El tamaño del IV es el correcto para asegurar la seguridad de las contraseñas (128 bits).
- El algoritmo usado es AES-256.
- Sólo la clave privada correcta permite el descifrado de la contraseña.
- Todos los datos de las contraseñas son ilegibles para una tercera persona.
- Al utilizarse una clave privada maestra segura y con el algoritmo AES-256, se asegura que los mensajes cifrados no son capaces de descifrarse en un tiempo razonable.

# Capítulo 4. DISCUSIÓN

En la realización del proyecto se han encontrado numerosas cuestiones que han tenido que ser resueltas para el correcto resultado de este, algunas más graves que otras. Pero finalmente, estas cuestiones se han sabido gestionar de manera correcta, aportando soluciones.

***Limitaciones del Estudio***

Una de las principales limitaciones del proyecto fue el uso de una blockchain privada. Si bien esto permitió un mayor control y personalización, también limitó la escalabilidad y la descentralización de la aplicación, al no haber tantos nodos como debería. Además, la necesidad de que los usuarios recuerden su clave privada maestra sin posibilidad de recuperación representó un desafío significativo en términos de usabilidad, aunque resultó la opción más óptima teniendo en cuenta el enfoque principal de la aplicación en cuanto a seguridad.

***Limitaciones de la Tecnología Empleada***

El uso de Firebase para la autenticación y Firestore para la base de datos proporcionó una solución rápida y eficiente, pero también presentó limitaciones en términos de flexibilidad y control sobre los datos. Por otro lado, la implementación de la blockchain fue una elección adecuada para el desarrollo inicial, aunque se identificaron ciertas limitaciones en el rendimiento y la capacidad de manejar grandes volúmenes de transacciones en tiempo real.

***Impacto del Proyecto***

El proyecto asegura que solo el usuario con la clave privada pueda acceder a las contraseñas, proporcionando alta seguridad y confidencialidad. Sin embargo, ha añadido complejidad al sistema y depende críticamente de que el usuario no pierda su clave privada. Además, hemos observado que puede afectar el rendimiento debido a los procesos de cifrado y descifrado. A pesar de estos desafíos, el proyecto ha demostrado ser una solución robusta y segura para la gestión de contraseñas, ofreciendo a los usuarios una herramienta confiable para proteger sus datos sensibles.

Aunque el proyecto enfrentó varias limitaciones y requirió ajustes a lo largo del desarrollo, los resultados finales son positivos y alineados con los objetivos principales. La experiencia adquirida y las lecciones aprendidas serán valiosas para futuros desarrollos en este campo.

# Capítulo 5. CONCLUSIONES

## 5.1 Conclusiones del trabajo

El objetivo general del proyecto era desarrollar una aplicación web de gestión de contraseñas basada en blockchain, asegurando que todas las contraseñas estén cifradas y sean legibles únicamente para el usuario con la clave privada maestra. Tras la implementación y evaluación, el proyecto cumplió con éxito este objetivo. La aplicación permite a los usuarios crear cuentas, iniciar sesión, gestionar sus contraseñas de manera segura utilizando una clave privada, generar contraseñas seguras automáticamente e informarse de cómo se guardan sus contraseñas en la blockchain.

La integración del frontend, backend y la blockchain privada para el almacenamiento seguro ha demostrado ser efectiva y funcional. Las pruebas realizadas validaron que el sistema cumple con los requisitos funcionales y de seguridad establecidos, ofreciendo una solución robusta y confiable para la gestión de contraseñas.

Por otro lado, la descentralización de la blockchain ofrece varias ventajas, como una mayor seguridad, ya que no depende de una entidad centralizada y cada nodo en la red verifica y almacena transacciones de manera independiente. Esto también promueve la transparencia y la inmutabilidad de los datos, aumentando la confianza de los usuarios.

Sin embargo, también presenta desventajas como la escalabilidad limitada y el consumo elevado de energía, especialmente cuando se utiliza la prueba de trabajo (Proof of Work). Además, la inmutabilidad de la blockchain puede dificultar la implementación de cambios, y la velocidad de las transacciones puede ser más lenta en comparación con sistemas centralizados.

## 5.2 Conclusiones personales

Desarrollar este proyecto ha sido una experiencia enriquecedora y desafiante. Desde el principio, me llamó la atención crear una aplicación que no solo fuese innovadora, sino también útil. La integración de las diferentes tecnologías no solo me permitió ampliar mis conocimientos técnicos, sino también mejorar mis habilidades en la gestión de proyectos y resolución de problemas.

Uno de los aprendizajes con más importancia durante la realización de la aplicación es la seguridad en el desarrollo de software, especialmente cuando se trata de gestionar datos sensibles como las contraseñas. La implementación de medidas de seguridad robustas y la constante validación del sistema fueron cruciales para el éxito del proyecto.

Además, este proyecto me ha permitido comprender mejor la importancia de la descentralización y cómo la blockchain puede ofrecer soluciones innovadoras a problemas comunes. La gestión de contraseñas es un tema crítico en la era digital, y contribuir con una solución que mejora la seguridad de los usuarios me ha brindado satisfacción personal.

Finalmente, trabajar en este proyecto me ha ayudado a desarrollar una visión más clara de mi futuro profesional, reafirmando mi interés en el desarrollo de aplicaciones basadas en tecnologías emergentes. La experiencia adquirida y las habilidades desarrolladas durante este proyecto serán fundamentales para mi carrera y mi crecimiento continuo en el campo de la informática.

# Capítulo 6. FUTURAS LÍNEAS DE TRABAJO

El proyecto desarrollado es un primer avance de lo que se podría realizar con esta idea en un futuro. Existen varias líneas de trabajo donde se puede continuar avanzando, dado que al ser una idea poco extendida en el mercado, se pueden incluir funcionalidades que hasta ahora no se han visto en la industria de gestores de contraseñas.

Cada nueva funcionalidad añadida puede ser la determinante para llamar la atención de la gran mayoría de usuarios.

Al comienzo del proyecto, se pensaron en algunas ideas ambiciosas, pero han estado limitadas por el tiempo y el objetivo de lo que es un trabajo fin de grado. Estas ideas, que se exponen a continuación, si bien no se han considerado para hacerlas en esta primera versión, pueden incluirse para hacerlo en las siguientes.

## 6.1 Blockchain privada

La aplicación actualmente está basada en una blockchain privada, personalizada para funcionar con las características específicas del proyecto, esto significa que los nodos de la blockchain deben estar gestionados por administradores de la aplicación, creando en muchos casos una descentralización no tan amplia.

Este enfoque se puede mejorar en un futuro adaptando la aplicación para funcionar con una blockchain pública ya establecida en el mercado y con muchos nodos descentralizados operativos, como por ejemplo Ethereum, una de las blockchains más descentralizadas que existen.

Esta idea también conlleva otro enfoque en cómo debe ser la seguridad, ya que todos los datos de todos los usuarios serán públicos a disposición de cualquier persona, aunque estos no sean legibles.

## 6.2 Base de datos centralizada

La base de datos centralizada Firebase en este proyecto es indispensable para el correcto funcionamiento de esta, pero también conlleva algunos problemas, como puede ser que la información que se guarda, que aunque no sea crítica, no está descentralizada. Además, cuantos más sistemas se utilicen más probabilidades hay de que alguno falle.

Esta base de datos en un futuro se podría suprimir, usando para todo la blockchain o incluso almacenamientos descentralizados existentes como *Nervos Network*. En cualquier caso, la autenticación que ahora proporciona Firebase, se podría cambiar por una opción parecida, pero basada en las carteras de una determinada blockchain. Es decir, que el usuario acceda a la aplicación sin necesidad de cuenta, solo conectando su cartera de criptomonedas que ya tiene.

## 6.3 Aplicación

La aplicación web desarrollada, a pesar de tener una interfaz intuitiva, se puede mejorar, quizás contando con ayuda profesional dedicada a este campo o simplemente dedicándole un tiempo mayor al destinado. Además, se puede añadir la funcionalidad multi-idioma, dado que actualmente todos los textos están traducidos al español, faltaría únicamente la lógica de intercambiarlos.

Por otro lado, el proyecto desarrollado está diseñado exclusivamente para interactuar a través de un navegador. Esto se podría cambiar desarrollando también la aplicación en un entorno móvil, realizando la plataforma en las formas nativas de los sistemas operativos móviles más demandados, Android e IOS.

# Capítulo 7. REFERENCIAS

Norton (2023). 139 password statistics to help you stay safe in 2023. Recuperado el 9 de mayo de 2024, de https://www.techopedia.com/es/estadisticas-ciberseguridad

Zippia. (2022). 20+ Essential Blockchain Statistics [2023]. Recuperado el 10 de mayo de 2024, de https://www.zippia.com/advice/blockchain-statistics/

Balmer, M. (2023). The Blockchain and blockchain development sector statistics and facts. Recuperado el 11 de mayo de 2024, de https://kruschecompany.com/blockchain-sector-statistics-and-facts/

Fernández, Y. (2024). Los 16 mejores gestores de contraseñas para proteger y recordar todas las que tengas. Xataka. Recuperado el 11 de mayo de 2024, de https://www.xataka.com/basics/16-mejores-gestores-contrasenas-para-proteger-recordar-todas-que-tengas

Wikipedia (2021). Historia de la criptografía (2024). Recuperado el 11 de mayo de 2024, de https://es.wikipedia.org/wiki/Historia_de_la_criptograf%C3%ADa#Criptograf%C3%ADa_moderna

EETimes. (2012). How Secure is AES Against Brute Force Attacks?. Recuperado el 11 de mayo de 2024, de https://www.eetimes.com/how-secure-is-aes-against-brute-force-attacks/

Kolesnikov, N. (2023). 50 Estadísticas Clave de Ciberseguridad para Mayo de 2024. Recuperado el 11 de mayo de 2024, de https://www.techopedia.com/es/estadisticas-ciberseguridad

Security.org (2023). Password Manager Industry Report and Market Outlook in 2023. Recuperado el 11 de mayo de 2024, de https://www.security.org/digital-safety/password-manager-annual-report/

FBI (2022). Internet Crime Report. Recuperado el 12 de mayo de 2024, de https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf

BOE (2018). Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.. Recuperado el 15 de mayo de 2024, de https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673

Parra, S. (2011). Almacenar las contraseñas en texto plano está prohibido. Recuperado el 15 de mayo de 2024, de https://www.eprivacidad.es/almacenar-contrasenas-texto-plano-prohibido/

Hive Systems. (2024). Are Your Passwords in the Green?. Recuperado el 19 de mayo de 2024, de https://www.hivesystems.com/blog/are-your-passwords-in-the-green.

José Sanchis. (2024). *Bitcoin Con Rigor*. Racks Labs.

Flaticon. (2024). Recuperado el 1 de junio de 2024, de https://www.flaticon.es/.

# Capítulo 8. ANEXOS

## 8.1 Estadísticas sobre la industria

A continuación se muestran unas estadísticas escogidas, las cuales han sido recopiladas de la página oficial de Norton y traducidas al castellano. Se pueden consultar más en su página web. (Norton, 2023)

1. El 27,5% de las personas dice que su contraseña más antigua tiene entre tres y cinco años. (Beyond Identity, 2021)
2. En promedio, se tarda 14 segundos en escribir una contraseña. (LastPass, 2021)
3. En 2022, más de 24 mil millones de contraseñas fueron expuestas por hackers. (Digital Shadows, 2022)
4. Más de 1 de cada 4 personas no sabe cuándo cambió por última vez la contraseña de su correo electrónico. (PC Matic, 2022)
5. El 64% de las contraseñas solo contienen entre ocho y once caracteres. (Security.org, 2023)
6. Solo el 10% de los consumidores informa haber usado una contraseña para iniciar sesión en sus cuentas de redes sociales en los últimos 60 días. (FIDO Alliance, 2022)
7. Casi el 60% de las personas fortalecen sus contraseñas como resultado de notar accesos no autorizados a sus cuentas o dispositivos. (Norton, 2021)
8. El 91% de las personas entiende que reutilizar contraseñas es un riesgo de seguridad. (LastPass, 2021)
9. Solo la mitad de los usuarios de internet están algo familiarizados con las mejores prácticas de seguridad de contraseñas. (Bitwarden, 2022)
10. El 69% de las veces, la Generación Z usa una variación de una sola contraseña. (LastPass, 2022)
11. Menos de la mitad de los estadounidenses creen firmemente que sus contraseñas son seguras. (Security.org, 2023)
12. 1 de cada 10 personas cree que alguien podría adivinar sus contraseñas solo mirando sus cuentas de redes sociales. (Beyond Identity, 2021)
13. Casi el 40% de las personas admite compartir sus contraseñas personales con otros. (Security.org, 2023)
14. El 89% de las personas sabe que usar la misma contraseña es un riesgo de seguridad, pero solo el 12% cambia las contraseñas entre cuentas. (LastPass, 2022)
15. Agregar un solo carácter especial a una contraseña común de 10 caracteres puede aumentar el tiempo que los hackers tardan en descifrarla en 1.5 horas. (Digital Shadows, 2022)
16. Solo el 12% de las personas siempre usa contraseñas únicas. (LastPass, 2022)
17. Casi el 40% de las personas no ha cambiado la contraseña de su Wi-Fi desde el día que la configuraron. (PC Matic, 2022)
18. Más de un tercio de las personas admite que se sentirían avergonzadas si tuvieran que leer su contraseña en voz alta. (Beyond Identity, 2021)

19. El 12% de las personas incluye el nombre de su pareja en sus contraseñas. (Security.org, 2023)
20. El 13% de las personas admite que pone el mismo nivel de esfuerzo en crear contraseñas, sin importar para qué tipo de cuenta sea. (LastPass, 2022)
21. El 61% de los afectados por el hacking de contraseñas tenían contraseñas de menos de ocho caracteres. (Security.org, 2023)
22. Menos del 50% de las personas cree que la contraseña de su cuenta de música en streaming es segura. (Beyond Identity, 2021)
23. Solo el 11% de los consumidores informa haber usado una contraseña para iniciar sesión en sus cuentas de streaming en los últimos 60 días. (FIDO Alliance, 2022)
24. El 10% de las personas ha usado la misma contraseña desde la escuela secundaria o preparatoria. (Beyond Identity, 2021)
25. El 50% de los líderes de TI creen que las contraseñas son una medida de seguridad demasiado débil. (Ping Identity, 2022)
26. El 2,2% de las personas informa usar una contraseña que tiene más de 21 años. (Beyond Identity, 2021)
27. Casi una cuarta parte de las personas dice que compartiría su contraseña con un compañero de cuarto. (Beyond Identity, 2021)
28. 45 millones de personas dependen de gestores de contraseñas para mantener el control de sus contraseñas. (Security.org, 2023)
29. 1 de cada 4 usuarios de internet reutiliza la contraseña maestra de su gestor de contraseñas para otras cuentas. (Security.org, 2023)
30. Casi el 35% de los usuarios de gestores de contraseñas afirma que su uso principal es asegurar que están usando contraseñas únicas en diferentes cuentas. (SANS, 2021)
31. La mitad de los usuarios de gestores de contraseñas lo usan solo para sus cuentas personales. (Security.org, 2023)
32. Casi 1 de cada 4 personas depende de un documento en su computadora para gestionar todas sus contraseñas. (Bitwarden, 2022)
33. 1 de cada 10 usuarios de gestores de contraseñas gasta entre $1 y $20 al año en su gestor de contraseñas. (Security.org, 2023)
34. Menos del 40% de las organizaciones requiere el uso de un gestor de contraseñas. (Ponemon Institute, 2020)
35. El 46% de los usuarios de gestores de contraseñas usa un gestor de contraseñas tanto para cuentas personales como laborales. (Security.org, 2023)
36. Aproximadamente 3 de cada 10 usuarios de gestores de contraseñas usan herramientas de gestión de contraseñas para asegurarse de tener fácil acceso a sus contraseñas. (SANS, 2021)

## 8.2 Métodos de la blockchain

Anteriormente se han enumerado los métodos de la blockchain y posteriormente explicado, a continuación se detalla el código de las mismas.

***Constructor (init)***

```
def __init__(self):
    self.chain = []
    self.data = []
    if os.path.exists('chain.json'):
        self.chain = self.upload_chain('chain.json')
    else:
        self.create_block(proof=1, previous_hash='0')
    self.nodes = set()
```

*Figura 42: Constructor de la clase blockchain*

***create_block()***

```
def create_block(self, proof, previous_hash):
    block = {'index': len(self.chain) + 1,
             'timestamp': str(datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')),
             'proof': proof,
             'previous_hash': previous_hash,
             'data': self.data}
    self.data = []
    self.chain.append(block)
    return block
```

*Figura 43: Método create_block de la clase blockchain*

***get_previous_block()***

```
def get_previous_block(self):
    return self.chain[-1]
```

*Figura 44: Método get_previous_block de la clase blockchain*

***proof_of_work()***

```
def proof_of_work(self, previous_proof):
    new_proof = 1
    check_proof = False
    while check_proof is False:
        hash_operation = hashlib.sha256(str(new_proof ** 2 - previous_proof ** 2).encode()).hexdigest()
        if hash_operation[:4] == '0000':
            check_proof = True
        else:
            new_proof += 1
    return new_proof
```

*Figura 45: Método proof_of_work de la clase blockchain*

***hash()***

```
def hash(self, block):
    encoded_block = json.dumps(block, sort_keys=True).encode()
    return hashlib.sha256(encoded_block).hexdigest()
```

*Figura 46: Método hash de la clase blockchain*

***is_chain_valid()***

```
def is_chain_valid(self, chain):
    previous_block = chain[0]
    block_index = 1
    while block_index < len(chain):
        block = chain[block_index]
        if block['previous_hash'] != self.hash(previous_block):
            return False
        previous_proof = previous_block['proof']
        proof = block['proof']
        hash_operation = hashlib.sha256(str(proof ** 2 - previous_proof ** 2).encode()).hexdigest()
        if hash_operation[:4] != '0000':
            return False
        previous_block = block
        block_index += 1
    return True
```

*Figura 47: Método is_chain_valid de la clase blockchain*

***add_data()***

```
def add_data(self, user, platform, key, salt):
    self.data.append({'user': user,
                      'platform': platform,
                      'key': key,
                      'iv': salt})
    previous_block = self.get_previous_block()
    return previous_block['index'] + 1
```

*Figura 48: Método add_data de la clase blockchain*

***add_user()***

```
def add_user(self, uid_user, encrypted_data, salt, iv):
    self.data.append({'uid_user': uid_user,
                      'encrypted_data': encrypted_data,
                      'salt': salt,
                      'iv': iv})
    previous_block = self.get_previous_block()
    return previous_block['index'] + 1
```

*Figura 49: Método add_user de la clase blockchain*

***add_node()***

```
def add_node(self, address):
    parsed_url = urlparse(address)
    self.nodes.add(parsed_url.netloc)
```

*Figura 50: Método add_node de la clase blockchain*

***get_data()***

```
def get_data(self, user, platform):
    blocks = self.chain
    for block in reversed(blocks):
        if block['data'] and 'user' in block['data'][0]:
            if block['data'][0]['user'] == user and block['data'][0]['platform'] == platform:
                return block
    return {'data': []}
```

*Figura 51: Método get_data de la clase blockchain*

***get_user()***

```
def get_user(self, uid_user):
    blocks = self.chain
    for block in reversed(blocks):
        if block['data'] and 'uid_user' in block['data'][0]:
            if block['data'][0]['uid_user'] == uid_user:
                return block
    return {'data': []}
```

*Figura 52: Método get_user de la clase blockchain*

***replace_chain()***

```
def replace_chain(self):
    network = self.nodes
    longest_chain = None
    max_length = len(self.chain)
    for node in network:
        response = requests.get(f'http://{node}/get_chain')
        if response.status_code == 200:
            length = response.json()['length']
            chain = response.json()['chain']
            if length > max_length and self.is_chain_valid(chain):
                max_length = length
                longest_chain = chain
    if longest_chain:
        self.chain = longest_chain
        return True
    return False
```

*Figura 53: Método replace_chain de la clase blockchain*

***save_chain()***

```
def save_chain(self, file_name):
    with open(file_name, 'w') as file:
        json.dump(self.chain, file, indent=4)
```

*Figura 54: Método save_chain de la clase blockchain*

***upload_chain()***

```
def upload_chain(self, file_name):
    with open(file_name, 'r') as file:
        chain = json.load(file)
    return chain
```

*Figura 55: Método upload_chain de la clase blockchain*

## 8.3 Cadena de bloques

La cadena de bloques se puede visualizar como un archivo JSON, en este caso este fichero se utiliza para inicializar la blockchain en caso de que exista.


```
[
    {
        "index": 1,
        "timestamp": "2024-05-05 20:26:58",
        "proof": 1,
        "previous_hash": "0",
        "data": []
    },
    {
        "index": 2,
        "timestamp": "2024-05-05 20:28:19",
        "proof": 533,
        "previous_hash": "e73e77d74d9752dc26646bbc153ab5a24310f65dbe0abd218d28d1c8d694f98b",
        "data": [
            {
                "uid_user":
"b27040d1832115e238bf394ca67609591a54c39ad08cd7700180640be70d7bf023f19bf85e4573be5757bbb26477
d7f06ff62cfd5c87c3a1ac874a390a6596f3",
                "encrypted_data":
"FC/57gh/awwoUcDDuQZd4ekoEbPFg3dPdJ7zpMUXFLVtRJ6Q3YfB+kyjrb9+
017hUDePfTGYbRgrbekTtPzLPabeXLDbU1C9osly2Zz2FAQ9ZVCdWEi3Ek+RFnaQb2wu/UXFTmMR7pkCvsq51EdvMt+1b
r4PJ0S5E5VMedHDrb185VmpRX37DZgHcaP3BLGQ",
                "salt": "e3d45580514ef6915a44d1f63c49656f",
                "iv": "6196a7d0e867bf8c1d8612d0fe606f71"
            }
        ]
    },
    {
        "index": 3,
        "timestamp": "2024-05-05 20:28:58",
        "proof": 45293,
        "previous_hash": "20a74e5996e9e5424d6875a1ed626aa081c28c29c598db7840e1dff781368a91",
        "data": [
            {
                "user":
"b27040d1832115e238bf394ca67609591a54c39ad08cd7700180640be70d7bf023f19bf85e4573be5757bbb26477
d7f06ff62cfd5c87c3a1ac874a390a6596f3",
                "platform":
"88633db2404442ad404c8cd806fe8d423f3702524de139c86aa76ef818469dee58bb620eb34844a24e997845f8b7
13c7189550f5d20204e389ef7160d45bff05",
                "key": "SQGskFYI3e4miGlM0p0qaA==",
                "iv": "96761f0fd832fc8ddc1b690567370ba2"
            }
        ]
    },
    {
        "index": 4,
        "timestamp": "2024-05-05 20:35:08",
        "proof": 21391,
        "previous_hash": "e6634bbc844ac93ba5c5751fa4b10b89d040931382151f460e1112dd5ad6cf75",
        "data": [
            {
                "user":
"b27040d1832115e238bf394ca67609591a54c39ad08cd7700180640be70d7bf023f19bf85e4573be5757bbb26477
d7f06ff62cfd5c87c3a1ac874a390a6596f3",
                "platform":
"0b7cd7cc373a61d6edd057f86f98abc75924423e087b3108fdeb1278a978b885882f2c31d21ed0eb79747e82184b
da5cc8fae6efe07f9272df4a9ad0a6b2a4a8",
                "key": "a21F8uuZN9md/N7Q+ZJiyA==",
                "iv": "0a977a53c287698541cc82e43e5b8d6c"
            }
        ]
    },
```


*Figura 56: Fragmento de la blockchain en JSON*

## 8.4 Casos de prueba

Lista detallada de los casos de prueba que se llevan a cabo, incluyendo:

- **Nombre + ID del Caso de Prueba**: Un identificador único para cada caso de prueba.
- **Tipo de Prueba + Entorno**: Prueba y entorno que se realizará.
- **Autor(es):** Persona que se encarga de la prueba
- **Descripción**: Una breve descripción de lo que se probará.
- **Precondiciones**: Cualquier condición que deba cumplirse antes de ejecutar el caso de prueba.
- **Instrucciones para la realización**: Pasos detallados para ejecutar el caso de prueba.
- **Recursos para realizar la prueba**: Datos específicos necesarios para la elaboración del caso de prueba.
- **Resultados Esperados**: El resultado esperado al ejecutar el caso de prueba.
- **Tiempo Estimado de Realización**: Tiempo que dura llevar a cabo la prueba.

A continuación se muestran los casos de prueba, incluyendo pruebas unitarias, de integración, rendimiento y seguridad.

***Llamar a la función de minar un bloque de la blockchain***

| **1. Nombre + Id caso de Prueba** | **Minar bloque de la blockchain - CT-001** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de la función de la blockchain de minar un bloque “mine_block()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/mine_block” con el método GET.<br>2. Pulsar el botón “Send” para enviar<br>3. Verificar que se devuelve un mensaje de éxito. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL “http://127.0.0.1:5000/mine_block” |
| 8. Resultado Esperado | La API debe responder con un mensaje de éxito donde se especifica los datos del bloque minado:<br>response = {'message': 'Felicidades, acabas de minar un bloque!',<br>'index': block['index'],<br>'timestamp': block['timestamp'],<br>'proof': block['proof'],<br>'previous_hash': block['previous_hash'],<br>'data': block['data']} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 14: Caso de prueba: Minar un bloque*

***Llamar a la función de obtener la cadena de la blockchain***

| **1. Nombre + Id caso de Prueba** | **Obtener la cadena de la blockchain - CT-002** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de la función de la obtener la cadena “get_chain()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/get_chain” con el método GET.<br>2. Pulsar el botón “Send” para enviar.<br>3. Verificar que se devuelve un mensaje con la cadena en JSON. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL “http://127.0.0.1:5000/get_chain” |
| 8. Resultado Esperado | La API debe responder con un mensaje devolviendo toda la cadena de la blockchain en formato JSON:<br>response = {'chain': blockchain.chain,<br>'length': len(blockchain.chain)} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 15: Caso de prueba: Obtener la cadena de la blockchain*

***Insertar los datos de una contraseña en la blockchain***

| **1. Nombre + Id caso de Prueba** | **Insertar los datos de una contraseña en la blockchain - CT-003** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de inserción de datos de una contraseña con la función “add_data()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/add_data” con el método POST.<br>3. Enviar los datos de la contraseña en formato JSON.<br>2. Pulsar el botón “Send” para enviar.<br>3. Verificar que se devuelve un mensaje con la cadena en JSON. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL “http://127.0.0.1:5000/add_data”<br>DATOS CONTRASEÑA:<br>{<br>"user": "peter",<br>"platform": "canvas",<br>"key": "peter321",<br>"iv": "o29shfHD62SF0ujfiWE"<br>} |
| 8. Resultado Esperado | La API debe responder con un mensaje de éxito:<br>response = {'message': f'La información ha sido añadida al Bloque {index}'} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 16: Caso de prueba: Insertar los datos de una contraseña en la blockchain*

***Insertar los datos de validación de clave privada en la blockchain***

| **1. Nombre + Id caso de Prueba** | **Insertar datos de validación de clave privada en la blockchain - CT-004** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de inserción de datos de validación de clave privada con la función “add_user()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/add_user” con el método POST.<br>3. Enviar los datos de la verificación en formato JSON.<br>2. Pulsar el botón “Send” para enviar.<br>3. Verificar que se devuelve un mensaje con la cadena en JSON. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL “http://127.0.0.1:5000/add_user”<br>DATOS DE VERIFICACIÓN:<br>{<br>"uid_user": "b27040d18321c87c3a1ac874a390a6596f3…",<br>"encrypted_data": ";4a390a6596fa6hy=",<br>"salt": "8321c87c3a1ac874a",<br>"iv": "27040d18321"<br>} |
| 8. Resultado Esperado | La API debe responder con un mensaje de éxito:<br>response = {'message': f'La información ha sido añadida al Bloque {index}'} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 17: Caso de prueba: Insertar los datos de validación de clave privada en la blockchain*

***Consultar los datos de una contraseña en la blockchain***

| **1. Nombre + Id caso de Prueba** | **Consultar los datos de una contraseña en la blockchain - CT-005** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de consulta de datos de una contraseña con la función “get_data()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/get_data?user=peter&platform=canvas” con el método GET.<br>2.Pulsar el botón “Send” para enviar.<br>3. Verificar que se devuelve un bloque con la contraseña o mensaje en JSON. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL:<br>“http://127.0.0.1:5000/get_data?user=peter&platform=canvas” |
| 8. Resultado Esperado | La API debe responder con un mensaje de éxito:<br>response = {'data': block}<br>O de que no se ha encontrado:<br>response = {'message': f'El usuario o la plataforma no existen'} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 18: Caso de prueba: Consultar los datos de una contraseña en la blockchain*

***Consultar los datos de validación de clave privada en la blockchain***

| **1. Nombre + Id caso de Prueba** | **Consultar datos de validación de clave privada en la blockchain - CT-006** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba Unitaria.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de consulta de datos de validación de clave privada con la función “get_user()”. Mediante la API de Flask. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman e insertar la ruta de la API “http://127.0.0.1:5000/get_user?uid_user=b27040d1832115e2…” con el método GET.<br>2. Pulsar el botón “Send” para enviar.<br>3. Verificar que se devuelve un bloque o mensaje en JSON. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: URL “http://127.0.0.1:5000/get_user?uid_user=b27040d18e2…” |
| 8. Resultado Esperado | La API debe responder con un mensaje de éxito:<br>response = {'data': block}<br>O de que no se ha encontrado:<br>response = {'message': 'El usuario no existe'} |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 19: Caso de prueba: Consultar los datos de validación de clave privada en la blockchain*

***Registrar usuario en la base de datos***

| **1. Nombre + Id caso de Prueba** | **Registrar usuario en la base de datos Firebase - CT-007** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de registro de un usuario en la base de datos Firebase. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La base de datos Firebase debe ser accesible y configurada correctamente para la autenticación y el Firestore. |
| 6. Instrucciones para la realización | 1. Abrir la aplicación web Keychain.<br>2. Pulsar el botón “Log In” y más seguidamente “Create account”.<br>3. Rellenar todos los campos y aceptar. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL “datos de Registro: User, email y contraseña”. |
| 8. Resultado Esperado | - La aplicación web debe redirigir hacia la página principal.<br>- El usuario está registrado en la plataforma Firebase. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 20: Caso de prueba: Registrar usuario en la base de datos Firebase*

***Iniciar sesión en la aplicación mediante la base de datos***

| **1. Nombre + Id caso de Prueba** | **Iniciar sesión en la aplicación mediante Firebase - CT-008** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de inicio de sesión de un usuario en la aplicación web mediante Firebase. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La base de datos Firebase debe ser accesible y configurada correctamente para la autenticación. |
| 6. Instrucciones para la realización | 1. Abrir la aplicación web Keychain.<br>2. Pulsar el botón “Log In”.<br>3. Rellenar todos los campos y aceptar. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL “datos de inicio de sesión: Email y contraseña” |
| 8. Resultado Esperado | - La aplicación web debe redirigir hacia la página principal. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 21: Caso de prueba: Iniciar sesión en la aplicación mediante Firebase*

***Verificar o registrar clave privada maestra***

| **1. Nombre + Id caso de Prueba** | **Verificar o registrar clave privada maestra en la blockchain mediante la aplicación web - CT-009** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento verificación y registro de la clave privada maestra. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R3: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R4: Haber iniciado sesión en la aplicación. |
| 6. Instrucciones para la realización | 1. Navegar a la pantalla de gestor de contraseñas.<br>2. Insertar clave privada maestra.<br>3. Pulsar el botón “Enter”. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL datos: Email y contraseña para inicio de sesión.<br>Clave privada maestra. |
| 8. Resultado Esperado | - La aplicación web debe redirigir hacia la ventana de gestionar las contraseñas .<br>- O por el contrario si es incorrecta la clave privada muestra un mensaje de error. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 22: Caso de prueba: Verificar o registrar clave privada maestra*

***Añadir contraseña en la aplicación y blockchain***

| **1. Nombre + Id caso de Prueba** | **Añadir contraseña en la aplicación y blockchain - CT-0010** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de la inserción de la contraseña en la blockchain. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R3: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R5: La base de datos Firebase debe ser accesible y configurada correctamente con Firestore.<br>- R6: Haber iniciado sesión en la aplicación.<br>- R7: Haber verificado la clave privada maestra. |
| 6. Instrucciones para la realización | 1. Navegar a la pantalla de gestor de contraseñas.<br>2. Insertar clave privada maestra.<br>3. Pulsar el botón “Enter”<br>4. Pulsar “añadir contraseña”.<br>5. Rellenar los campos y aceptar. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL datos: Email y contraseña para inicio de sesión.<br>Clave privada maestra.<br>Plataforma y contraseña para insertar. |
| 8. Resultado Esperado | - Se crea un bloque con la contraseña cifrada en la blockchain.<br>- Se añade el nombre de la plataforma a la base de datos Firestore.<br>- La aplicación web debe redirigir hacia la ventana de gestionar las contraseñas refrescando para ver la nueva contraseña insertada.<br>- por el contrario dar mensaje de error por: No introducir bien la contraseña y su repetición o intentar repetir una plataforma ya registrada. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 23: Caso de prueba: Añadir contraseña en la aplicación y blockchain*

***Modificar o eliminar contraseña en la aplicación y blockchain***

| **1. Nombre + Id caso de Prueba** | **Modificar o eliminar contraseña en la aplicación y blockchain - CT-011** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de la inserción de la contraseña en la blockchain. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R3: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R4: La base de datos Firebase debe ser accesible y configurada correctamente con Firestore.<br>- R5: Haber iniciado sesión en la aplicación.<br>- R6: Haber verificado la clave privada maestra.<br>- R7: Tener al menos una contraseña registrada. |
| 6. Instrucciones para la realización | 1. Navegar a la pantalla de gestor de contraseñas.<br>2. Insertar clave privada maestra.<br>3. Pulsar el botón “Modificar” en la contraseña deseada.<br>4. Cambiar los campos requeridos y aceptar. O pulsar eliminar. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL datos: Email y contraseña para inicio de sesión.<br>Clave privada maestra.<br>Nueva plataforma o contraseña en el caso de editar contraseña. |
| 8. Resultado Esperado | - El nuevo dato debe estar registrado en un nuevo bloque.<br>- En el caso de eliminar, debe desaparecer el nombre de la plataforma de la base de datos Firestore.<br>- La aplicación web debe redirigir hacia la ventana de gestionar las contraseñas refrescando para ver la nueva contraseña insertada.<br>- O por el contrario dar mensaje de error por: No cambiar ningún campo o intentar repetir una plataforma ya registrada. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 24: Caso de prueba: Modificar o eliminar contraseña en la aplicación y blockchain*

***Consultar contraseña cifradas de la blockchain***

| **1. Nombre + Id caso de Prueba** | **Consultar contraseña cifradas de la blockchain - CT-012** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Integración.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar el correcto funcionamiento de consulta de contraseñas cifradas en la blockchain. |
| 5. Precondiciones | - R1: La aplicación web debe estar operativa.<br>- R2: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R3: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R4: La base de datos Firebase debe ser accesible y configurada correctamente con Firestore.<br>- R5: Haber iniciado sesión en la aplicación.<br>- R6: Haber verificado la clave privada maestra.<br>- R7: Tener al menos una contraseña registrada. |
| 6. Instrucciones para la realización | 1. Navegar a la pantalla de gestor de contraseñas.<br>2. Insertar clave privada maestra.<br>3. Pulsar el botón “consultar mis datos en blockchain”. |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener un navegador operativo para acceder a la aplicación web.<br>- Datos: URL datos: Email y contraseña para inicio de sesión.<br>Clave privada maestra. |
| 8. Resultado Esperado | La aplicación web debe redirigir hacia la ventana de consultar contraseñas, mostrando en el caso de que hay alguna contraseña registrada, las mismas almacenadas en formato JSON y cifradas. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 25: Caso de prueba: Consultar contraseña cifradas de la blockchain*

***Verificar tiempo de respuesta de la API***

| **1. Nombre + Id caso de Prueba** | **Verificar tiempo de respuesta de la API - CT-013** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Rendimiento.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar que el tiempo de respuesta de la API está dentro de los límites aceptables. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener descargada y abierta la aplicación Postman. |
| 6. Instrucciones para la realización | 1. Abrir Postman y probar todos los endpoints de la API.<br>2. Apuntar el tiempo de respuesta resultante en Postman.<br>3. Realizar una media de los tiempos de respuesta |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Obtener la aplicación Postman y tener acceso al servidor donde se encuentra la API.<br>- Datos: Endpoints” |
| 8. Resultado Esperado | - La API debe tener como mínimo un tiempo de respuesta bueno, entre 100 ms y 300 ms. |
| 9. Tiempo Estimado de Realización | 5 minutos. |

*Tabla 26: Caso de prueba: Verificar tiempo de respuesta de la API*

***Verificación de Seguridad del Cifrado***

| **1. Nombre + Id caso de Prueba** | **Verificar robustez del cifrado de contraseñas - CT-014** |
|---|---|
| 2. Tipo de Prueba + Entorno | Tipo de Prueba: Prueba de Seguridad.<br>Entorno: Ambiente de Desarrollo (simulación realista) |
| 3. Autor(es) | Yago Iglesias Díaz. |
| 4. Descripción | Resumen: Verificar que el cifrado de las contraseñas almacenadas en la blockchain es robusto y seguro. |
| 5. Precondiciones | - R1: La red de nodos debe estar operativa, al menos un nodo en localhost.<br>- R2: El servidor debe estar operativo para alojar la API que conecta con la blockchain.<br>- R3: Tener acceso a los bloques de la blockchain. |
| 6. Instrucciones para la realización | 1. Consultar el bloque con la información de una contraseña cifrada.<br>2. Comprobar que los siguientes campos están en el bloque: *user, platform, key* e *IV*.<br>3. Comprobar la longitud en bits de los campos *key* e *IV*<br>4. Comprobar que todos los campos son ilegibles.<br>5. Verificar que la clave de desencriptado, descifra correctamente la variable *'key'.* |
| 7. Recursos para realizar la prueba | - Recursos Técnicos: Tener acceso a los datos de la blockchain.<br>- Datos: Un bloque de la blockchain con los datos de una contraseña y la clave de descifrado. |
| 8. Resultado Esperado | - Algoritmo de Cifrado: Debe ser el algoritmo seguro AES-256.<br>- Longitud de Claves y IVs: Las claves de cifrado deben ser de 256 bits y el mensaje cifrado y el IV de 128 bits.<br>- Resistencia al Ataque: Los mensajes cifrados no deben ser capaces de descifrarse en un tiempo razonable.<br>- Integridad del Bloque: Todos los campos cifrados deben permanecer cifrados o hasheados.<br>- Pruebas de Descifrado: Sólo la clave privada correcta debe permitir el descifrado de la contraseña. |
| 9. Tiempo Estimado de Realización | 10 minutos por cada bloque a probar. |

*Tabla 27: Caso de prueba: Verificación de Seguridad del Cifrado*